МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Российский государственный гуманитарный университет»
(ФГБОУ ВО «РГГУ»)

ИНСТИТУТ ИНФОРМАЦИОННЫХ НАУК И ТЕХНОЛОГИЙ БЕЗОПАСНОСТИ
ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ СИСТЕМ И БЕЗОПАСНОСТИ
Кафедра информационных технологий и систем

УПРАВЛЕНИЕ ОБЛАЧНЫМИ ИНФОРМАЦИОННЫМИ РЕСУРСАМИ В ГУМАНИТАРНОЙ СФЕРЕ

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

09.03.03 Прикладная информатика
Код и наименование направления подготовки

Прикладная информатика в гуманитарной сфере
Наименование направленности (профиля)

Уровень высшего образования: *бакалавриат*

Форма обучения: *очная*

РПД адаптирована для лиц
с ограниченными возможностями
здоровья и инвалидов

Москва 2023

Управление облачными информационными ресурсами в гуманитарной сфере
Рабочая программа дисциплины
Составитель:
к.т.н., доцент Е. Б. Карелина

……..
Ответственный редактор
к.с.-х.н., доц., зав. кафедрой Н.Ш. Шукенбаева

……

УТВЕРЖДЕНО
Протокол заседания кафедры ИТС
№ 8 от 15.04.2023г.

ОГЛАВЛЕНИЕ

1. Пояснительная записка

1.1. Цель и задачи дисциплины

Цель дисциплины: теоретическое и практическое освоение основ облачных вычислений: понятие облака, программное обеспечение и аппаратные средства облачных вычислений, архитектуры облачных приложений и модели облачных инфраструктур, мобильные приложения, основы облачной обработки данных, подготовка к переходу на облачные вычисления, обеспечение безопасности данных в облаке, масштабирование облачной инфраструктуры.

Задачи:

- освоить общие принципы работы с облачной инфраструктурой и приложениями;
- раскрыть особенности использования различных моделей развертывания облачных инфраструктур;
- ознакомиться с архитектурами облачных приложений: технология Grid Computing, транзакционные вычисления;
- ознакомиться с принципами облачной обработки данных на базе решений различных фирм;
- познакомиться с процессом подготовки к переходу на облачные вычисления;
- раскрыть особенности, связанные с обеспечением данных в облаке.

1.2. Перечень планируемых результатов обучения по дисциплине, соотнесенных с индикаторами достижения компетенций

Компетенция	Индикаторы компетенций	Результаты обучения
ПК-6 Способен настраивать, эксплуатировать и сопровождать информационные системы и сервисы	ПК-6.1 Знает методы настройки, порядок и мероприятия по эксплуатации и сопровождению информационных систем и сервисов	**Знать:** преимущества облачной инфраструктуры. **Уметь:** работать с различными облачными сервисами как единолично, так и в команде. **Владеть:** навыками работы с комплектами средств анализа требований к мобильным облачным приложения.
	ПК-6.2 Умеет организовывать настройку, эксплуатацию и сопровождение информационных	**Знать:** отличие различных моделей развертывания облачных инфраструктур **Уметь:** анализировать требования к облачным сервисам.

	систем и сервисов	**Владеть:** навыками работы с комплектами средств проектирования мобильных облачных приложений.
	ПК-6.3 Владеет навыками управления конфигурацией ИС и сервисов в процессе эксплуатации, решения проблем и консультирования пользователей информационных систем и сервисов	**Знать:** принципы облачной обработки данных **Уметь:** проектировать облачные сервисы как единолично, так и в команде. **Владеть:** навыками работы с комплектами средств разработки мобильных облачных приложений.
ПК-8 Способен принимать участие в организации ИТ-инфраструктуры и управлении информационной безопасностью	ПК-8.1 Знает способы организации ИТ-инфраструктуры, методы и приемы управления информационной безопасностью	**Знать:** принципы создания мобильных приложений для работы в облаке **Уметь:** разрабатывать облачные сервисы как единолично, так и в команде. **Владеть:** навыками работы с комплектами средств настройки мобильных облачных приложений.
	ПК-8.2 Умеет организовывать ИТ-инфраструктуру предприятия и процессы управления информационной безопасностью	**Знать:** структуру процесса перехода на облачные вычисления **Уметь:** отлаживать облачные сервисы как единолично, так и в команде. **Владеть:** навыками работы с комплектами средств поддержки мобильных облачных приложений.
	ПК-8.3 Владеет навыками организации ИТ-инфраструктуры и управления информационной безопасностью	**Знать:** способы обеспечения защиты информации в облаке. **Уметь:** сопровождать облачные сервисы как единолично, так и в команде. **Владеть:** навыками работы с комплектами средств подбора проектных решений для мобильных облачных приложений.

1.3. Место дисциплины в структуре образовательной программы

Дисциплина «Управление облачными информационными ресурсами в гуманитарной сфере» относится к части, формируемой участниками образовательных отношений блока дисциплин учебного плана.

Для освоения дисциплины необходимы знания, умения и владения, сформированные в ходе изучения дисциплин Программирование C#, Программирование C++, Программирование Java, Архитектура вычислительных систем, Базы данных, Информационно-поисковые системы и машины в гуманитарной сфере, Методы информационного поиска в задачах информатизации гуманитарной сферы.

В результате освоения дисциплины формируются знания, умения и владения, необходимые для изучения следующих дисциплин и прохождения практик: прохождения преддипломной практики.

2. Структура дисциплины

Общая трудоёмкость дисциплины составляет 3 з.е., 108 академических часов.

Структура дисциплины для очной формы обучения

Объем дисциплины в форме контактной работы обучающихся с педагогическими работниками и (или) лицами, привлекаемыми к реализации образовательной программы на иных условиях, при проведении учебных занятий:

Семестр	Тип учебных занятий	Количество часов
8	Лекции	14
8	Практические занятия	28
	Всего:	42

Объем дисциплины (модуля) в форме самостоятельной работы обучающихся составляет 66 академических часов.

3. Содержание дисциплины

№	Наименование раздела дисциплины	Содержание
1.	Тема 1. Тенденции развития современных инфраструктурных решений	Рассматриваются основные этапы развития аппаратного и программного обеспечения. Проводится небольшой исторический обзор. Рассматриваются основные современные тенденции развития аппаратного обеспечения, основные требования к инфраструктуре. Рассматриваются современные тенденции развития инфраструктурных решений, которые привели к появлению концепции облачных вычислений
2.	Тема 2. Общие представления о технологиях виртуализации	Рассматривается технология виртуализации, которая занимает ключевое место в концепции "облачных" вычислений.
3.	Тема 3. Основы облачных вычислений	Cloud Computing, понятие Software as a Service (SaaS)
4.	Тема 4. Облачные Веб-службы	Рассматриваются некоторые из веб-служб, предоставляемые концепцией облачных вычислений. "Инфраструктура как Сервис" (Infrastructure-as-a-Service, IaaS), "Коммуникаций как Сервис" (Communication-as-a-Service, CaaS). "Программное обеспечение как Сервис" (Software-as-a-Service, SaaS). Ключевые особенности использования внешних источников/ресурсов (outsourcing), доступные как "Платформы как Сервис" (Platforms-as-a-Service, PaaS).

5.	Тема 5. Введение в Windows Azure SDK	Рассматриваются основные возможности Windows Azure SDK.

4. Образовательные технологии

Для проведения учебных занятий по дисциплине используются различные образовательные технологии. Для организации учебного процесса может быть использовано электронное обучение и (или) дистанционные образовательные технологии.

5. Оценка планируемых результатов обучения

5.1 Система оценивания

Форма контроля	**Макс. количество баллов**	
	За одну работу	**Всего**
Текущий контроль: Практическая работа № 1, защита отчета Практическая работа № 2, защита отчета Практическая работа № 3, защита отчета Доклад № 1 по выбранной теме Доклад № 2 по выбранной теме	 16 баллов 16 баллов 16 баллов 6 баллов 6 баллов	 16 баллов 16 баллов 16 баллов 6 баллов 6 баллов
Промежуточная аттестация *зачет с оценкой*		40 баллов
Итого за семестр		100 баллов

Полученный совокупный результат конвертируется в традиционную шкалу оценок и в шкалу оценок Европейской системы переноса и накопления кредитов (European Credit Transfer System; далее – ECTS) в соответствии с таблицей:

100-балльная шкала	Традиционная шкала		Шкала ECTS
95 – 100	отлично	зачтено	A
83 – 94			B
68 – 82	хорошо		C
56 – 67	удовлетворительно		D
50 – 55			E
20 – 49	неудовлетворительно	не зачтено	FX
0 – 19			F

5.2 Критерии выставления оценки по дисциплине

Баллы/ Шкала ECTS	**Оценка по дисциплине**	**Критерии оценки результатов обучения по дисциплине**
100-83/ A,B	отлично/ зачтено	Выставляется обучающемуся, если он глубоко и прочно усвоил теоретический и практический материал, может продемонстрировать это на занятиях и в ходе промежуточной аттестации. Обучающийся исчерпывающе и логически стройно излагает учебный материал, умеет увязывать теорию с практикой, справляется с решением задач профессиональной направленности высокого уровня сложности, правильно обосновывает принятые решения. Свободно ориентируется в учебной и профессиональной литературе.

Баллы/ Шкала ECTS	Оценка по дисциплине	Критерии оценки результатов обучения по дисциплине
		Оценка по дисциплине выставляются обучающемуся с учётом результатов текущей и промежуточной аттестации. Компетенции, закреплённые за дисциплиной, сформированы на уровне – «высокий».
82-68/ C	хорошо/ зачтено	Выставляется обучающемуся, если он знает теоретический и практический материал, грамотно и по существу излагает его на занятиях и в ходе промежуточной аттестации, не допуская существенных неточностей. Обучающийся правильно применяет теоретические положения при решении практических задач профессиональной направленности разного уровня сложности, владеет необходимыми для этого навыками и приёмами. Достаточно хорошо ориентируется в учебной и профессиональной литературе. Оценка по дисциплине выставляются обучающемуся с учётом результатов текущей и промежуточной аттестации. Компетенции, закреплённые за дисциплиной, сформированы на уровне – «хороший».
67-50/ D,E	удовлетво-рительно/ зачтено	Выставляется обучающемуся, если он знает на базовом уровне теоретический и практический материал, допускает отдельные ошибки при его изложении на занятиях и в ходе промежуточной аттестации. Обучающийся испытывает определённые затруднения в применении теоретических положений при решении практических задач профессиональной направленности стандартного уровня сложности, владеет необходимыми для этого базовыми навыками и приёмами. Демонстрирует достаточный уровень знания учебной литературы по дисциплине. Оценка по дисциплине выставляются обучающемуся с учётом результатов текущей и промежуточной аттестации. Компетенции, закреплённые за дисциплиной, сформированы на уровне – «достаточный».
49-0/ F,FX	неудовлет-ворительно/ не зачтено	Выставляется обучающемуся, если он не знает на базовом уровне теоретический и практический материал, допускает грубые ошибки при его изложении на занятиях и в ходе промежуточной аттестации. Обучающийся испытывает серьёзные затруднения в применении теоретических положений при решении практических задач профессиональной направленности стандартного уровня сложности, не владеет необходимыми для этого навыками и приёмами. Демонстрирует фрагментарные знания учебной литературы по дисциплине. Оценка по дисциплине выставляются обучающемуся с учётом результатов текущей и промежуточной аттестации. Компетенции на уровне «достаточный», закреплённые за дисциплиной, не сформированы.

5.3 Оценочные средства (материалы) для текущего контроля успеваемости, промежуточной аттестации обучающихся по дисциплине

1. Основные характеристики облачных вычислений?
2. Отличия серверных и «облачных» технологий?
3. Предпосылки перехода в «облака»?
4. Основные «облачных» архитектуры?
5. Основные характеристики IaaS?
6. Основные характеристики SaaS?
7. Основные характеристики PaaS?
8. Основные риски, связанные с использованием облачных вычислений?
9. Архитектуры публичных «облаков»?
10. Архитектуры приватных «облаков»?
11. Архитектуры гибридных «облаков»?
12. Экземпляр облачного приложения. Состояние приложения. Жизненный цикл.

13. Хранение пользовательских данных в «облаке»?
14. Хранение данных приложения в «облаке»?
15. Реляционные и нереляционные облачные БД?
16. Среды разработки и фреймворки для разработки облачных сервисов?
17. Инструменты эмуляции работы в «облаке» на локальном компьютере?
18. Основные компоненты платформы Amazon EC2?
19. Основные компоненты платформы Google Apps?
20. Основные компоненты платформы Windows Azure?
21. Что такое «мультитенантность»?
22. Благодаря чему достигается масштабируемость облачных сервисов?
23. Благодаря чему достигается 100% время доступности облачных сервисов?
24. Способы хранения данных в Windows Azure?
25. Образы операционных систем доступные в Amazon EC2?
26. Из чего складывается цена размещения приложения на платформе Google Apps?
27. Область применения гибридных «облаков»?
28. Основные ограничения при использовании публичных «облаков», связанные с законодательными и нормативными данными, действующими на территории РФ?

Тематика рефератов (докладов)

1. Основные характеристики облачных вычислений?
2. Отличия серверных и «облачных» технологий?
3. Предпосылки перехода в «облака»?
4. Основные «облачных» архитектуры?
5. Основные характеристики IaaS?
6. Основные характеристики SaaS?
7. Основные характеристики PaaS?
8. Основные риски, связанные с использованием облачных вычислений?
9. Архитектуры публичных «облаков»?
10. Архитектуры приватных «облаков»?
11. Архитектуры гибридных «облаков»?
12. Экземпляр облачного приложения. Состояние приложения. Жизненный цикл.
13. Хранение пользовательских данных в «облаке»?
14. Хранение данных приложения в «облаке»?
15. Реляционные и нереляционные облачные БД?
16. Среды разработки и фреймворки для разработки облачных сервисов?
17. Инструменты эмуляции работы в «облаке» на локальном компьютере?
18. Основные компоненты платформы Amazon EC2?
19. Основные компоненты платформы Google Apps?
20. Основные компоненты платформы Windows Azure?
21. Что такое «мультитенантность»?
22. Благодаря чему достигается масштабируемость облачных сервисов?
23. Благодаря чему достигается 100% время доступности облачных сервисов?
24. Способы хранения данных в Windows Azure?
25. Образы операционных систем доступные в Amazon EC2?
26. Из чего складывается цена размещения приложения на платформе Google Apps?
27. Область применения гибридных «облаков»?
28. Основные ограничения при использовании публичных «облаков», связанные с законодательными и нормативными данными, действующими на территории РФ?
29. Что нельзя хранить в публичном «облаке» в России?
30. Что нельзя хранить в публичном «облаке» в США?

6. Учебно-методическое и информационное обеспечение дисциплины

6.1 Список источников и литературы

Источники
Основные

1. Федеральный закон Российской Федерации от 27 июля 2006 г. N 149-ФЗ «Об информации, информационных технологиях и о защите информации».
2. ГОСТ 34.003-90. Автоматизированные системы. Термины и определения.
3. ГОСТ 34.201-89. Информационная технология. Комплекс стандартов на автоматизированные системы. Виды, комплектность и обозначение документов при создании автоматизированных систем.
4. ГОСТ 34.601-90. Автоматизированные системы. Комплекс стандартов на автоматизированные системы. Стадии создания.

Литература

Основная

1. Блюмин, А. М. Мировые информационные ресурсы : учебное пособие для бакалавров / А. М. Блюмин, Н. А. Феоктистов. - 4-е изд., стер. - Москва : Издательско-торговая корпорация «Дашков и К°», 2020. - 382 с. - ISBN 978-5-394-03598-2. - Текст : электронный. - URL: https://znanium.com/catalog/product/1093525.
2. Эрджиес, К. Распределенные системы реального времени: теория и практика : практическое руководство / К. Эрджиес ; пер. с анг. В. А. Яроцкий. - Москва : ДМК Пресс, 2020. - 382 с. - ISBN 978-5-97060-852-4. - Текст : электронный. - URL: https://znanium.com/catalog/product/1210669.
3. Бабичев С.Л., Коньков К.А. Распределенные системы: учебное пособие для вузов. – Москва: Издательство Юрайт, 2020. – 507 с. – (Высшее образование). URL: https://urait.ru/viewer/raspredelennye-sistemy-457005#page/2

Дополнительная

1. Резник, В. Г. Распределенные сервис-ориентированные системы : учебное пособие / В. Г. Резник. - Томск : Томский государственный университет систем управления и радиоэлектроники, 2020. - 305 с. - Текст : электронный. - URL: https://znanium.com/catalog/product/1845905.
2. Федотова, Е. Л. Информационные технологии в науке и образовании : учебное пособие / Е.Л. Федотова, А.А. Федотов. — Москва : ФОРУМ : ИНФРА-М, 2023. — 335 с. — (Высшее образование). - ISBN 978-5-8199-0884-6. - Текст : электронный. - URL: https://znanium.com/catalog/product/1891636.
3. Синаторов, С. В. Информационные технологии : учебное пособие / С. В. Синаторов. - 2-е изд., стер. - Москва : Флинта, 2021. - 448 с. - ISBN 978-5-9765-1717-2. - Текст : электронный. - URL: https://znanium.com/catalog/product/1304012.
4. Дружинин, Д. В. Высокопроизводительные вычисления и облачные технологии : учебное пособие / Д. В. Дружинин. - Томск : Издательство Томского государственного университета, 2020. - 94 с. - ISBN 978-5-94621-921-1. - Текст : электронный. - URL: https://znanium.com/catalog/product/1864757.

6.2 Перечень ресурсов информационно-телекоммуникационной сети «Интернет».

Национальная электронная библиотека (НЭБ) www.rusneb.ru

ELibrary.ru Научная электронная библиотека www.elibrary.ru
Электронная библиотека Grebennikon.ru www.grebennikon.ru
Cambridge University Press

6.3 Профессиональные базы данных и информационно-справочные системы

Доступ к профессиональным базам данных: https://liber.rsuh.ru/ru/bases

Информационные справочные системы:
1. Консультант Плюс
2. Гарант

7. Материально-техническое обеспечение дисциплины

№п/п	Наименование специальных помещений и помещений для самостоятельной работы	Оснащенность специальных помещений и помещений для самостоятельной работы	Перечень лицензионного программного обеспечения. Реквизиты подтверждающего документа		
			Наименование ПО	Лицензия/сертификат/заказ	Дата лицензии
1.	Лаборатория информатики – ауд. № 203	1 компьютер преподавателя, 12 компьютеров обучающихся, маркерная доска, проектор	Windows 7 Microsoft office 2010 Pro Microsoft Visual Professional 2019 Mozilla Firefox 52.8.1 ESR Matlab Mathcad Education - University edition Kaspersky Endpoint Security	68526624 49420326 63202190 свободный доступ 647526 2996385 17E0-181226-094912-873-979	без даты 08.12.2011 без даты свободный доступ без даты 14.06.2019 26.12.2018

8. Обеспечение образовательного процесса для лиц с ограниченными возможностями здоровья и инвалидов

В ходе реализации дисциплины используются следующие дополнительные методы обучения, текущего контроля успеваемости и промежуточной аттестации обучающихся в зависимости от их индивидуальных особенностей:

- для слепых и слабовидящих: лекции оформляются в виде электронного документа, доступного с помощью компьютера со специализированным программным обеспечением; письменные задания выполняются на компьютере со специализированным программным обеспечением или могут быть заменены устным ответом; обеспечивается индивидуальное равномерное освещение не менее 300 люкс; для выполнения задания при необходимости предоставляется увеличивающее устройство; возможно также использование собственных увеличивающих устройств; письменные задания оформляются увеличенным шрифтом; экзамен и зачёт проводятся в устной форме или выполняются в письменной форме на компьютере.

- для глухих и слабослышащих: лекции оформляются в виде электронного документа, либо предоставляется звукоусиливающая аппаратура индивидуального пользования; письменные задания выполняются на компьютере в письменной форме; экзамен и зачёт проводятся в письменной форме на компьютере; возможно проведение в форме тестирования.
- для лиц с нарушениями опорно-двигательного аппарата: лекции оформляются в виде электронного документа, доступного с помощью компьютера со специализированным программным обеспечением; письменные задания выполняются на компьютере со специализированным программным обеспечением; экзамен и зачёт проводятся в устной форме или выполняются в письменной форме на компьютере.

При необходимости предусматривается увеличение времени для подготовки ответа.

Процедура проведения промежуточной аттестации для обучающихся устанавливается с учётом их индивидуальных психофизических особенностей. Промежуточная аттестация может проводиться в несколько этапов.

При проведении процедуры оценивания результатов обучения предусматривается использование технических средств, необходимых в связи с индивидуальными особенностями обучающихся. Эти средства могут быть предоставлены университетом, или могут использоваться собственные технические средства.

Проведение процедуры оценивания результатов обучения допускается с использованием дистанционных образовательных технологий.

Обеспечивается доступ к информационным и библиографическим ресурсам в сети Интернет для каждого обучающегося в формах, адаптированных к ограничениям их здоровья и восприятия информации:

- для слепых и слабовидящих: в печатной форме увеличенным шрифтом, в форме электронного документа, в форме аудиофайла.
- для глухих и слабослышащих: в печатной форме, в форме электронного документа.
- для обучающихся с нарушениями опорно-двигательного аппарата: в печатной форме, в форме электронного документа, в форме аудиофайла.

Учебные аудитории для всех видов контактной и самостоятельной работы, научная библиотека и иные помещения для обучения оснащены специальным оборудованием и учебными местами с техническими средствами обучения:

- для слепых и слабовидящих: устройством для сканирования и чтения с камерой SARA CE; дисплеем Брайля PAC Mate 20; принтером Брайля EmBraille ViewPlus;
- для глухих и слабослышащих: автоматизированным рабочим местом для людей с нарушением слуха и слабослышащих; акустический усилитель и колонки;
- для обучающихся с нарушениями опорно-двигательного аппарата: передвижными, регулируемыми эргономическими партами СИ-1; компьютерной техникой со специальным программным обеспечением.

9. Методические материалы

9.1 Планы практических занятий

В плане практических занятий выполняются следующие работы;

Практическая работа 1. Создание первого приложения для windows azure

Задача 1 – создание проекта в Visual Studio

Задача 2 – Создание модели данных для работы с табличным хранилищем

Задача 3 – создание веб-роли, позволяющей отображать содержимое гостевой книги и добавлять записи

Задача 4 – использование очередей для организации фоновой обработки данных

Практическая работа 2. Фоновая обработка данных с использованием прикладной роли и очереди

Задача 1 – создание прикладной роли для фоновой обработки данных

Практическая работа 3. Развертывание приложения в windows azure

Задача 1 – создание сервиса хранения данных и вычислительного сервиса

Задача 2 – развертывание приложения с помощью портала Windows Azure

Задача 3 – изменение числа экземпляров роли

Задача 4 – проверка приложения в тестовой среде

Задача 5 – передача приложения в промышленную среду

Практическая работа 1. Создание первого приложения для windows azure

Задача 1 – создание проекта в Visual Studio

В данной задаче вы создадите новый проект типа Cloud Service (облачный сервис).

1. Откройте Visual Studio от имени административной учетной записи: выберите пункты меню **Start | All Programs | Microsoft Visual Studio 2010**, щелкните правой кнопкой мыши на ярлыке **Microsoft Visual Studio 2010** и выберите пункт **Run as administrator**.
2. В случае появления диалога **User Account Control** нажмите кнопку **Continue**.
3. В меню **File** выберите пункт **New**, затем **Project**.
4. В диалоге **New Project** разверните в списке **Installed Templates** соответствующий предпочитаемому вами языку узел (Visual C# или Visual Basic) и выберите пункт **Cloud**. Выберите шаблон проекта **Windows Azure Project**, присвойте проекту имя (поле **Name)** – в нашем случае **GuestBook**, установите расположение (поле **Location**) в **Source\Ex1-BuildingYourFirstWindowsAzureApp\[CS|VB]** в подкаталоге в каталоге с материалами, измените имя решения (**Solution name**) на **GuestBook**, и убедитесь в том, что установлен флажок **Create directory for solution** (Создать каталог для решения). Нажмите кнопку **OK** для создания проекта.

В диалоге **New Windows Azure Project** на панели **Roles** раскройте закладку, соответствующую предпочитаемому вами языку (Visual C# или Visual Basic), выберите из списка тип роли **ASP.NET Web Role** и нажмите кнопку с указывающей вправо стрелкой (>),

чтобы добавить экземпляр роли данного типа в решение. Перед тем, как закрыть данный диалог, выберите добавленную роль в правой панели, нажмите на пиктограмму с изображением карандаша и переименуйте роль в **GuestBook_WebRole**. Нажмите кнопку **OK** для создания решения.

В обозревателе решения (**Solution Explorer**) отобразится структура созданного решения.

Задача 2 – Создание модели данных для работы с табличным хранилищем

Приложение хранит записи гостевой книги в табличном хранилище Windows Azure. Табличное хранилище предоставляет частично структурированные таблицы, содержащие наборы сущностей. Каждая сущность содержит первичный ключ и набор типизированных свойств в формате свойство-значение.

Дополнительно к используемым вашей моделью свойствам каждая сущность в табличном хранилище должна иметь два ключевых свойства: **PartitionKey** и **RowKey**. Вместе эти два свойства формируют первичный ключ и уникально идентифицируют каждую сущность в таблице. Кроме того, каждая сущность имеет системное свойство **Timestamp**, используемое службой для хранения даты последних изменений. Данное поле предназначено для служебного использования и не может изменяться внутри приложения. Программный интерфейс Table Storage client API предоставляет класс **TableServiceEntity**, в котором уже определены необходимые свойства. Хотя вы можете унаследовать от него ваш собственный класс, это не является обязательным условием.

Программный интерфейс Table service API совместим с предоставляемым технологией WCF Data Services (ранее она называлась ADO.NET Data Services Framework) протоколом REST API, что позволяет использовать библиотеку WCF Data Services Client Library (ранее она называлась .NET Client Library) и работать с табличным хранилищем с использованием объектов .NET. Табличное хранилище не содержит какой-либо информации о схеме таблиц, что позволяет хранить в одной таблице сущности с различными наборами свойств. В то же время этот подход не является рекомендованным и в демонстрационном приложении GuestBook в таблицах хранятся данные с фиксированной схемой.

Чтобы работать с данными в хранилище с использованием WCF Data Services Client Library, необходимо определить класс, наследующий от **TableServiceContext**, в свою очередь

являющийся потомком описанного в WCF Data Services класса **DataServiceContext**. Интерфейс Table Storage API позволяет приложениям создавать таблицы на основании описанных контекстных классов. Чтобы это стало возможным, каждый контекстный класс должен публиковать таблицы как свойства типа **IQueryable<*SchemaClass*>**, где ***SchemaClass*** – конкретный класс, описывающий хранящуюся в таблице сущность.
В этой задаче вы создадите схему для хранения сущностей, которая потребуется для приложения GuestBook. Затем вы создадите классы, которые позволят использовать WCF Data Services для доступа к табличному хранилищу. В завершение задачи вы создадите объект, который будет использоваться элементами управления ASP.NET и предоставит функциональность чтения, обновления и удаления записей.
1. Создайте проект для размещения в нем классов схемы. Для этого в меню **File** выберите пункты **Add**, затем **New Project**.
2. В диалоге **Add New Project** в панели **Installed Templates** раскройте соответствующий предпочитаемому языку узел, выберите категорию **Windows**, укажите шаблон проекта **Class Library**. Измените имя на **GuestBook_Data**, не меняя предлагаемое расположение, и нажмите **OK**.

3. Удалите сгенерированный класс. Для этого щелкните правой кнопкой на файле **Class1.cs** (для проекта на Visual C#) или **Class1.vb** (для проекта на Visual Basic) и выберите **Delete**. Нажмите кнопку **OK** в диалоге подтверждения.
4. Добавьте в проект **GuestBook_Data** ссылку на библиотеку .NET Client Library for WCF Data Services. На панели **Solution Explorer** щелкните правой кнопкой на узле проекта **GuestBook_Data**, выберите **Add Reference**, перейдите на закладку **.NET**, выберите компонент **System.Data.Services.Client** и нажмите **OK**.

5. Повторите предыдущий шаг и добавьте ссылку на сборку Windows Azure storage client API, для чего выберите компонент **Microsoft.WindowsAzure.StorageClient**.
6. Перед тем, как сохранять сущность в таблице, для нее необходимо описать схему. Чтобы сделать это, щелкните в панели **Solution Explorer** правой кнопкой на проекте **GuestBook_Data**, выберите **Add**, затем **Class**. В диалоге **Add New Item** укажите имя - **GuestBookEntry.cs** (для проекта на Visual C#) или **GuestBookEntry.vb** (для проекта на Visual Basic) и нажмите кнопку **Add**.

7. В верхней части файла подключите пространство имен **Microsoft.WindowsAzure.StorageClient**.

```
using Microsoft.WindowsAzure.StorageClient;
```
Visual Basic
```
Imports Microsoft.WindowsAzure.StorageClient
```
8. Откройте файл **GuestBookEntry.cs** (для проекта на Visual C#) или **GuestBookEntry.vb** (для проекта на Visual Basic) и обновите объявление класса **GuestBookEntry**, сделав его открытым и наследующим от класса **TableServiceEntity**.

C#
```
public class GuestBookEntry
: Microsoft.WindowsAzure.StorageClient.TableServiceEntity
{
}
```
Visual Basic
```
Public Class GuestBookEntry
Inherits Microsoft.WindowsAzure.StorageClient.TableServiceEntity
```

End Class

9. Добавьте в класс **GuestBookEntry** конструктор, инициализирующий свойства **PartitionKey** и **RowKey**.
(Фрагмент кода – *Introduction to Windows Azure - Ex1 GuestBookEntry constructor – CS*)
C#
```
public GuestBookEntry()
{
PartitionKey = DateTime.UtcNow.ToString("MMddyyyy");

// Row key allows sorting, so we make sure the rows come back in time order.
RowKey = string.Format("{0:10}_{1}", DateTime.MaxValue.Ticks - DateTime.Now.Ticks,
Guid.NewGuid());
}
```
(Фрагмент кода – *Introduction to Windows Azure - Ex1 GuestBookEntry constructor – VB*)
Visual Basic
```
Public Sub New()
PartitionKey = DateTime.UtcNow.ToString("MMddyyyy")
' Row key allows sorting, so we make sure the rows come back in time order.
RowKey = String.Format("{0:10}_{1}", DateTime.MaxValue.Ticks - DateTime.Now.Ticks,
Guid.NewGuid())
End Sub
```

10. Чтобы завершить описание класса **GuestBookEntry**, добавьте свойства **Message**, **GuestName**, **PhotoUrl** и **ThumbnailUrl**, хранящие полезную нагрузку.
(Фрагмент кода – *Introduction to Windows Azure - Ex1 Table Schema Properties – CS*)
C#
```
public string Message { get; set; }
public string GuestName { get; set; }
public string PhotoUrl { get; set; }
public string ThumbnailUrl { get; set; }
```


(Фрагмент кода – *Introduction to Windows Azure - Ex1 Table Schema Properties – VB*)

11. Сохраните файл **GuestBookEntry.cs** (для проекта на Visual C#) или **GuestBookEntry.vb** (для проекта на Visual Basic).
12. Далее необходимо создать класс, позволяющий взаимодействовать с табличным хранилищем с использованием WCF Data Services. Чтобы сделать это, на панели **Solution Explorer** щелкните правой кнопкой на проекте **GuestBook_Data**, выберите **Add**, затем **Class**. В диалоге **Add New Item** установите имя (поле **Name)** в **GuestBookDataContext.cs** (для проекта на Visual C#) или **GuestBookDataContext.vb** (для проекта на Visual Basic) и нажмите **Add**.
13. В созданном классе обновите его объявление таким образом, чтобы он стал открытым и наследовал от класса **TableServiceContext**.

C#
```
public class GuestBookDataContext
: Microsoft.WindowsAzure.StorageClient.TableServiceContext
{
```

```
}
```
Visual Basic
```
Public Class GuestBookDataContext
Inherits Microsoft.WindowsAzure.StorageClient.TableServiceContext
End Class
```
14. Теперь добавьте конструктор, инициализирующий базовый класс информацией о реквизитах для доступа к табличному хранилищу.
(Фрагмент кода – *Introduction to Windows Azure - Ex1 GuestBookDataContext Class – CS*)
C#
```
public class GuestBookDataContext
: Microsoft.WindowsAzure.StorageClient.TableServiceContext

{
public GuestBookDataContext(string baseAddress,
Microsoft.WindowsAzure.StorageCredentials credentials)
: base(baseAddress, credentials)
{ }
}
```

Visual Basic
```
Public Class GuestBookDataContext
Inherits Microsoft.WindowsAzure.StorageClient.TableServiceContext
Public Sub New(ByVal baseAddress As String, ByVal credentials As
Microsoft.WindowsAzure.StorageCredentials)
MyBase.New(baseAddress, credentials)
End Sub
End Class
```

15. Добавьте в класс **GuestBookDataContext** свойство для таблицы сущностей типа **GuestBookEntry**. Чтобы сделать это, вставьте следующий (выделенный) код в класс.
(Фрагмент кода – *Introduction to Windows Azure - Ex1 GuestBookEntry Property – CS*)
C#
```
public class GuestBookDataContext
: Microsoft.WindowsAzure.StorageClient.TableServiceContext
{
...
public IQueryable<GuestBookEntry> GuestBookEntry
{
get
{
return this.CreateQuery<GuestBookEntry>("GuestBookEntry");
}

}
}
```

Visual Basic
```
Public Class GuestBookDataContext
Inherits Microsoft.WindowsAzure.StorageClient.TableServiceContext
...
Public ReadOnly Property GuestBookEntry() As IQueryable(Of GuestBookEntry)
Get
```

```
Return Me.CreateQuery(Of GuestBookEntry)("GuestBookEntry")
End Get
End Property
End Class
```

16. Теперь вам понадобится объект, который может быть использован для привязки данных в компонентах ASP.NET. На панели **Solution Explorer** щелкните правой кнопкой на проекте **GuestBook_Data**, выберите **Add**, затем **Class**. В диалоге **Add New Item** измените имя на **GuestBookDataSource.cs** (для проекта на Visual C#) или **GuestBookDataSource.vb** (для проекта на Visual Basic) и нажмите **Add**.
17. Во вновь добавленный класс импортируйте пространства имен **Microsoft.WindowsAzure** и **Microsoft.WindowsAzure.StorageClient**.

C#
```
using Microsoft.WindowsAzure;
using Microsoft.WindowsAzure.StorageClient;
```
Visual Basic
```
Imports Microsoft.WindowsAzure
Imports Microsoft.WindowsAzure.StorageClient
```

18. Сделайте класс **GuestBookDataSource** открытым (**public**), а также добавьте в него свойства для класса контекста и информации об учетной записи хранилища, как показано ниже.

C#
```
public class GuestBookDataSource
{
private static CloudStorageAccount storageAccount;
private GuestBookDataContext context;
}
```

Visual Basic
```
Public Class GuestBookDataSource
Private Shared storageAccount As CloudStorageAccount
Private context As GuestBookDataContext
End Class
```

19. Теперь добавьте статический конструктор (Shared в Visual Basic), как показано в следующем фрагменте кода (подсвеченная часть). Код в конструкторе создает таблицы на основе описания класса **GuestBookDataContext**.

C#
```
public class GuestBookDataSource
{
...
static GuestBookDataSource()
{
storageAccount = CloudStorageAccount.FromConfigurationSetting("DataConnectionString");

CloudTableClient.CreateTablesFromModel(
typeof(GuestBookDataContext),
storageAccount.TableEndpoint.AbsoluteUri,
storageAccount.Credentials);
}
```

```
}
```

Visual Basic

```
Public Class GuestBookDataSource
...
Shared Sub New()
storageAccount = CloudStorageAccount.FromConfigurationSetting("DataConnectionString")
CloudTableClient.CreateTablesFromModel(GetType(GuestBookDataContext),
storageAccount.TableEndpoint.AbsoluteUri, storageAccount.Credentials)
End Sub
End Class
```

20. Добавьте в класс **GuestBookDataSource** конструктор, инициализирующий использующийся для доступа к данным класс.
(Фрагмент кода – *Introduction to Windows Azure - Ex1 GuestBookDataSource Constructor – CS*)

C#

```
public class GuestBookDataSource
{
...
public GuestBookDataSource()

{
this.context = new GuestBookDataContext(storageAccount.TableEndpoint.AbsoluteUri,
storageAccount.Credentials);
this.context.RetryPolicy = RetryPolicies.Retry(3, TimeSpan.FromSeconds(1));
}
}
```

Visual Basic

```
Public Class GuestBookDataSource
...
Public Sub New()
Me.context = New GuestBookDataContext(storageAccount.TableEndpoint.AbsoluteUri,
storageAccount.Credentials)
Me.context.RetryPolicy = RetryPolicies.Retry(3, TimeSpan.FromSeconds(1))
End Sub
End Class
```

21. Теперь добавьте метод, возвращающий содержимое таблицы *GuestBookEntry*.
(Фрагмент кода – *Introduction to Windows Azure - Ex1 GuestBookDataSource Select – CS*)

C#

```
public class GuestBookDataSource
{
...
public IEnumerable<GuestBookEntry> GetGuestBookEntries()
{
var results = from g in this.context.GuestBookEntry
where g.PartitionKey == DateTime.UtcNow.ToString("MMddyyyy")
select g;
return results;
}
```

```
}
```

Visual Basic

```
Public Class GuestBookDataSource
...
Public Function GetGuestBookEntries() As IEnumerable(Of GuestBookEntry)
Dim results = From g In Me.context.GuestBookEntry _
Where g.PartitionKey = DateTime.UtcNow.ToString("MMddyyyy") _
Select g
Return results
End Function
End Class
```

22. Теперь добавьте метод, добавляющий запись в таблицу *GuestBookEntry*.

C#

```
public class GuestBookDataSource
{
...
public void AddGuestBookEntry(GuestBookEntry newItem)
{
this.context.AddObject("GuestBookEntry", newItem);
this.context.SaveChanges();
}
}
```

Visual Basic

```
Public Class GuestBookDataSource

...
Public Sub AddGuestBookEntry(ByVal newItem As GuestBookEntry)
Me.context.AddObject("GuestBookEntry", newItem)
Me.context.SaveChanges()
End Sub
End Class
```

23. В заключение добавьте метод, обновляющий URL изображения-миниатюры.

C#

```
public class GuestBookDataSource
{
...
public void UpdateImageThumbnail(string partitionKey, string rowKey, string thumbUrl)
{
var results = from g in this.context.GuestBookEntry
where g.PartitionKey == partitionKey && g.RowKey == rowKey
select g;
var entry = results.FirstOrDefault<GuestBookEntry>();
entry.ThumbnailUrl = thumbUrl;
this.context.UpdateObject(entry);
```

this.context.SaveChanges();
}
}

24. Сохраните файла **GuestBookDataSource.cs** (для проекта на Visual C#) или **GuestBookDataSource.vb** (для проекта на Visual Basic).

Задача 3 – создание веб-роли, позволяющей отображать содержимое гостевой книги и добавлять записи

В этой задаче вы доработаете созданную в задаче 1 веб-роль. Изменения затронут пользовательский интерфейс, после чего он сможет отображать содержимое гостевой книги. Вы не будете настраивать содержимое страницы вручную, вместо этого возьмете существующую страницу, находящуюся в каталоге **Assets** с материалами к данному упражнению. Далее вы добавите код, сохраняющий сущности в таблице, а изображения – в хранилище двоичных объектов.

1. Добавьте в веб-роль ссылку на проект **GuestBook_Data**. В панели **Solution Explorer** щелкните правой кнопкой на узле проекта **GuestBook_WebRole**, выберите **Add Reference**, переключитесь на закладку **Projects**, после чего выберите проект **GuestBook_Data** и нажмите кнопку **OK**.

2. При создании веб-роли была сгенерирована страница **Default.aspx**. Вы замените ее другой, предварительно подготовленной для вас. Чтобы удалить страницу, в панели **Solution Explorer** щелкните правой кнопкой на файле **Default.aspx** в проекте **GuestBook_WebRole** и выберите **Delete**.

3. Добавьте предварительно подготовленную страницу в веб-роль. Чтобы сделать это, щелкните правой кнопкой на проекте **GuestBook_WebRole** в **Solution Explorer**, выберите **Add** | **Existing Item**. В диалоге **Add Existing Item** перейдите в каталог **Source\Ex1-BuildingYourFirstWindowsAzureApp\Assets**, выберите соответствующий предпочитаемому языку каталог (Visual C# или Visual Basic); удерживая кнопку **CTRL**, выберите все файлы и нажмите кнопку **Add**.

4. Откройте файл кода для главной страницы проекта **GuestBook_WebRole**. Чтобы сделать это, щелкните правой кнопкой на файле **Default.aspx** и выберите пункт **View Code**.

5. Добавьте объявления следующих пространств имен.

(Фрагмент кода – *Introduction to Windows Azure - Ex1 Web Role Namespace Declarations – CS*)

C#
using System.IO;
using System.Net;
using Microsoft.WindowsAzure;
using Microsoft.WindowsAzure.ServiceRuntime;
using Microsoft.WindowsAzure.StorageClient;
using GuestBook_Data;

6. Добавьте в класс **Default** объявления свойств.

(Фрагмент кода – *Introduction to Windows Azure - Ex1 Web Role Member Fields – CS*)

C#
public partial class Default : System.Web.UI.Page
{
private static bool storageInitialized = false;
private static object gate = new Object();
private static CloudBlobClient blobStorage;
...

```
}
```

7. Добавьте обработчик события **SignButton_Click** со следующим содержимым.
(Фрагмент кода – *Introduction to Windows Azure - Ex1 SignButton_Click – CS*)
C#

```
public partial class Default : System.Web.UI.Page
{
...
protected void SignButton_Click(object sender, EventArgs e)
{
if (FileUpload1.HasFile)
{
InitializeStorage();

// upload the image to blob storage
string uniqueBlobName = string.Format("guestbookpics/image_{0}{1}",
Guid.NewGuid().ToString(), Path.GetExtension(FileUpload1.FileName));
CloudBlockBlob blob = blobStorage.GetBlockBlobReference(uniqueBlobName);
blob.Properties.ContentType = FileUpload1.PostedFile.ContentType;
blob.UploadFromStream(FileUpload1.FileContent);
System.Diagnostics.Trace.TraceInformation("Uploaded image '{0}' to blob storage as '{1}'",
FileUpload1.FileName, uniqueBlobName);
// create a new entry in table storage
GuestBookEntry entry = new GuestBookEntry() { GuestName = NameTextBox.Text, Message =
MessageTextBox.Text, PhotoUrl = blob.Uri.ToString(), ThumbnailUrl = blob.Uri.ToString() };
GuestBookDataSource ds = new GuestBookDataSource();
ds.AddGuestBookEntry(entry);
System.Diagnostics.Trace.TraceInformation("Added entry {0}-{1} in table storage for guest
'{2}'", entry.PartitionKey, entry.RowKey, entry.GuestName);
}
NameTextBox.Text = "";
MessageTextBox.Text = "";
DataList1.DataBind();
}
}
```

8. Обновите содержимое метода **Timer1_Tick** в соответствии с приведенным ниже текстом.
(Фрагмент кода – *Introduction to Windows Azure - Ex1 Timer1_Tick – CS*)
C#

```
public partial class Default : System.Web.UI.Page
{
...
protected void Timer1_Tick(object sender, EventArgs e)
{
DataList1.DataBind();
}
}
```

9. Найдите обработчик события **Page_Load** и обновите его содержимое в соответствие с приведенным ниже примером, чтобы включить таймер.

C#

```
public partial class Default : System.Web.UI.Page
{
...
protected void Page_Load(object sender, EventArgs e)
{
if (!Page.IsPostBack)
{
Timer1.Enabled = true;
}
}
}
```

10. Добавьте реализацию метода **InitializeStorage** в соответствии с приведенным ниже фрагментом (выделенная часть кода).
(Фрагмент кода – *Introduction to Windows Azure - Ex1 InitializeStorage – CS*)

C#

```
public partial class Default : System.Web.UI.Page
{
...
private void InitializeStorage()

{
if (storageInitialized)
{
return;
}
lock (gate)
{
if (storageInitialized)
{
return;
}
try
{
// read account configuration settings
var storageAccount =
CloudStorageAccount.FromConfigurationSetting("DataConnectionString");
// create blob container for images
blobStorage = storageAccount.CreateCloudBlobClient();
CloudBlobContainer container = blobStorage.GetContainerReference("guestbookpics");
container.CreateIfNotExist();
// configure container for public access
var permissions = container.GetPermissions();
permissions.PublicAccess = BlobContainerPublicAccessType.Container;
container.SetPermissions(permissions);
}
catch (WebException)
{
throw new WebException("Storage services initialization failure. "
+ "Check your storage account configuration settings. If running locally, "
```

```
+ "ensure that the Development Storage service is running.");
}
storageInitialized = true;
}
}
}
```

11. Поскольку веб-роль использует сервисы хранения данных Windows Azure (Windows Azure storage services), необходимо хранить реквизиты для подключения к ним. Чтобы создать новую настройку, в панели **Solution Explorer** разверните узел **Roles**, находящийся в проекте **GuestBook** и дважды щелкните на роли **GuestBook_WebRole**. В настройках роли перейдите на закладку **Settings**. Нажмите **Add Setting**, укажите в качестве имени строку "*DataConnectionString*" (колонка **Name**), измените тип на *ConnectionString*, после чего нажмите кнопку с многоточием.

12. В диалоге **Storage Account Connection String** выберите вариант **Use the Windows Azure storage emulator** (использовать эмулируемое хранилище) и нажмите **OK**.

13. Нажмите сочетание клавиш **CTRL + S**, чтобы сохранить изменения в файле конфигурации.
14. Теперь необходимо предусмотреть, что будет служить в качестве источника информации о конфигурации. В проекте **GuestBook_WebRole** откройте файл **Global.asax.cs** (для проекта на Visual C#) или **Global.asax.vb** (для проекта на Visual Basic).

15. Добавьте объявления пространств имен Microsoft.WindowsAzure и Microsoft.WindowsAzure.ServiceRuntime.

C#

```
using Microsoft.WindowsAzure;
using Microsoft.WindowsAzure.ServiceRuntime;
```

16. Добавьте в метод **Application_Start** следующий фрагмент кода.
(Фрагмент кода – *Introduction to Windows Azure - Ex1 SetConfigurationSettingPublisher – CS*)

C#

```
void Application_Start(object sender, EventArgs e)
{
Microsoft.WindowsAzure.CloudStorageAccount.SetConfigurationSettingPublisher((configName
, configSetter) =>
{
configSetter(RoleEnvironment.GetConfigurationSettingValue(configName));
});
}
```

Задача 4 – использование очередей для организации фоновой обработки данных

В этом упражнении вы обновите веб-роль таким образом, чтобы она помещала в очередь требующие фоновой обработки рабочие элементы. Эти элементы будут находиться в очереди до
тех пор, пока не будут извлечены из нее прикладной ролью. Прикладная роль извлекает из очереди очередной элемент и создает миниатюру для каждого добавленного пользователем изображения.

1. Откройте файл кода для файла **Default.aspx**. Для этого щелкните на нем правой кнопкой мыши и выберите **View Code**.
2. Добавьте в класс **Default** экземпляр класса **CloudQueueClient**, используемый для взаимодействия с очередью (выделенный фрагмент).
(Фрагмент кода – *Introduction to Windows Azure - Ex1 CloudQueueClient member – CS*)

C#

```
public partial class _Default : System.Web.UI.Page
{
private static bool storageInitialized = false;
private static object gate = new Object();
private static CloudBlobClient blobStorage;
private static CloudQueueClient queueStorage;
...
}
```

3. Теперь обновите инициализирующий хранилище код таким образом, чтобы он дополнительно создавал очередь, если не существует, и инициализировал добавленную выше переменную. Чтобы сделать это, найдите метод **InitializeStorage** и добавьте в него приведенный ниже код (выделенный фрагмент) сразу после кода, конфигурирующего контейнер двоичных объектов.

C#

```
public partial class Default : System.Web.UI.Page
{
...
private void InitializeStorage()
{
...
try
{
...
// configure container for public access
var permissions = container.GetPermissions();
permissions.PublicAccess = BlobContainerPublicAccessType.Container;
container.SetPermissions(permissions);
// create queue to communicate with worker role
queueStorage = storageAccount.CreateCloudQueueClient();
CloudQueue queue = queueStorage.GetQueueReference("guestthumbs");
queue.CreateIfNotExist();
}
catch (WebException)
{
...
}
```

4. В завершение добавьте код, помещающий рабочие элементы в очередь. Найдите обработчик нажатия **SignButton_Click** и вставьте в него приведенный ниже код (выделенный фрагмент) сразу после добавления новой сущности в табличное хранилище.

(Фрагмент кода – *Introduction to Windows Azure - Ex1 Queueing work items – CS*)

C#

```
protected void SignButton_Click(object sender, EventArgs e)
{
if (FileUpload1.HasFile)
{
...
// create a new entry in table storage
GuestBookEntry entry = new GuestBookEntry() { GuestName = NameTextBox.Text, Message =
MessageTextBox.Text, PhotoUrl = blob.Uri.ToString(), ThumbnailUrl = blob.Uri.ToString() };
```

```
GuestBookDataSource ds = new GuestBookDataSource();
ds.AddGuestBookEntry(entry);

System.Diagnostics.Trace.TraceInformation("Added entry {0}-{1} in table storage for guest '{2}'",
entry.PartitionKey, entry.RowKey, entry.GuestName);
// queue a message to process the image
var queue = queueStorage.GetQueueReference("guestthumbs");
var message = new CloudQueueMessage(String.Format("{0},{1},{2}", blob.Uri.ToString(),
entry.PartitionKey, entry.RowKey));
queue.AddMessage(message);
System.Diagnostics.Trace.TraceInformation("Queued message to process blob '{0}'",
uniqueBlobName);
}
NameTextBox.Text = "";
MessageTextBox.Text = "";
DataList1.DataBind();
}
```

Практическая работа 2. Фоновая обработка данных с использованием прикладной роли и очереди

В этой задаче вы добавите прикладную роль в решение и доработаете ее таким образом, чтобы она могла извлекать сообщения из очереди и обрабатывать их.

1. Запустите Visual Studio от имени администратора, если это не было сделано ранее: выберите пункт меню **Start | All Programs | Microsoft Visual Studio 2010**. Щелкните правой кнопкой на ярлыке **Microsoft Visual Studio 2010** и выберите элемент **Run as administrator**.
2. В меню **File** выберите **Open | Project/Solution**. В диалоге **Open Project** перейдите в подкаталог **Source\Ex2-UsingWorkerRolesAndQueues\Begin** в папке с материалами к работе и укажите файл **Begin.sln** в соответствующем предпочитаемому языку каталоге (Visual C# или Visual Basic), затем нажмите **Open**. Вы также можете продолжить работу с решением, полученным после завершения предыдущего шага.
3. В панели **Solution Explorer** нажмите правой кнопкой на узле **Roles** проекта **GuestBook**, далее выберите **Add | New Worker Role Project**.
4. В диалоге **Add New Role Project** выберите категорию **Worker Role** и шаблон **Worker Role** для предпочитаемого языка (Visual C# или Visual Basic). Измените имя добавленной роли на **GuestBook_WorkerRole** и нажмите **Add**.

5. В новом проекте добавьте ссылку на модель данных. Для этого в панели **Solution Explorer** нажмите правой кнопкой на проекте **GuestBook_WorkerRole**, выберите пункт **Add Reference**, переключитесь на закладку **Projects**, выберите **GuestBook_Data** и нажмите кнопку **OK**.
6. Теперь добавьте ссылку на сборку **System.Drawing**, для чего в диалоге **Add Reference** переключитесь на закладку **.NET**, выберите компонент **System.Drawing** и нажмите **OK**.
7. Откройте файл **WorkerRole.cs** (для проекта на Visual C#) или **WorkerRole.vb** (для проекта на Visual Basic) из проекта **GuestBook_WorkerRole** и добавьте объявления следующих пространств имен.

(Фрагмент кода – *Introduction to Windows Azure - Ex2 WorkerRole Namespaces – CS*)

C#

```
using System.Drawing;
using System.Drawing.Drawing2D;
using System.Drawing.Imaging;
using System.IO;
```

using GuestBook_Data;

8. Добавьте в класс **WorkerRole** экземпляры классов для работы с двоичным хранилищем и очередью.
(Фрагмент кода – *Introduction to Windows Azure - Ex2 WorkerRole Fields – CS*)
C#

```
public class WorkerRole : RoleEntryPoint
{
private CloudQueue queue;
private CloudBlobContainer container;
...
}
```

9. Добавьте приведенный ниже фрагмент в метод **OnStart** перед вызовом метода **OnStart** базового класса.
(Фрагмент кода – *Introduction to Windows Azure - Ex2 WorkerRole OnStart – CS*)
C#

```
public class WorkerRole : RoleEntryPoint
{
...
public override bool OnStart()

{
// Set the maximum number of concurrent connections
ServicePointManager.DefaultConnectionLimit = 12;
// read storage account configuration settings
CloudStorageAccount.SetConfigurationSettingPublisher((configName, configSetter) =>
{
configSetter(RoleEnvironment.GetConfigurationSettingValue(configName));
});
var storageAccount =
CloudStorageAccount.FromConfigurationSetting("DataConnectionString");
// initialize blob storage
CloudBlobClient blobStorage = storageAccount.CreateCloudBlobClient();
container = blobStorage.GetContainerReference("guestbookpics");
// initialize queue storage
CloudQueueClient queueStorage = storageAccount.CreateCloudQueueClient();
queue = queueStorage.GetQueueReference("guestthumbs");
Trace.TraceInformation("Creating container and queue...");
bool storageInitialized = false;
while (!storageInitialized)
{
try
{
// create the blob container and allow public access
container.CreateIfNotExist();
var permissions = container.GetPermissions();
permissions.PublicAccess = BlobContainerPublicAccessType.Container;
container.SetPermissions(permissions);

// create the message queue(s)
queue.CreateIfNotExist();
```

```
storageInitialized = true;
}
catch (StorageClientException e)
{
if (e.ErrorCode == StorageErrorCode.TransportError)
{
Trace.TraceError("Storage services initialization failure. "
+ "Check your storage account configuration settings. If running locally, "
+ "ensure that the Development Storage service is running. Message: '{0}'", e.Message);
System.Threading.Thread.Sleep(5000);
}
else
{
throw;
}
}
}

return base.OnStart();
}
...
}
```

10. Замените тело метода **Run** приведенным ниже фрагментом.
(Фрагмент кода – *Introduction to Windows Azure - Ex2 WorkerRole Run – CS*)

C#

```
public class WorkerRole : RoleEntryPoint
{
...
public override void Run()
{
Trace.TraceInformation("Listening for queue messages...");
while (true)
{
try
{
// retrieve a new message from the queue
CloudQueueMessage msg = queue.GetMessage();
if (msg != null)
{

// parse message retrieved from queue
var messageParts = msg.AsString.Split(new char[] { ',' });
var imageBlobUri = messageParts[0];
var partitionKey = messageParts[1];
var rowkey = messageParts[2];
Trace.TraceInformation("Processing image in blob '{0}'.", imageBlobUri);
string thumbnailBlobUri = System.Text.RegularExpressions.Regex.Replace(imageBlobUri,
"([^\\.]+)(\\.[^\\.]+)?$", "$1-thumb$2");
CloudBlob inputBlob = container.GetBlobReference(imageBlobUri);
CloudBlob outputBlob = container.GetBlobReference(thumbnailBlobUri);
using (BlobStream input = inputBlob.OpenRead())
```

```
using (BlobStream output = outputBlob.OpenWrite())
{
ProcessImage(input, output);
// commit the blob and set its properties
output.Commit();
outputBlob.Properties.ContentType = "image/jpeg";
outputBlob.SetProperties();
// update the entry in table storage to point to the thumbnail
GuestBookDataSource ds = new GuestBookDataSource();
ds.UpdateImageThumbnail(partitionKey, rowkey, thumbnailBlobUri);
// remove message from queue
queue.DeleteMessage(msg);
Trace.TraceInformation("Generated thumbnail in blob '{0}'.", thumbnailBlobUri);

}
}
else
{
System.Threading.Thread.Sleep(1000);
}
}
catch (StorageClientException e)
{
Trace.TraceError("Exception when processing queue item. Message: '{0}'", e.Message);
System.Threading.Thread.Sleep(5000);
}
}
}
...
}
```

11. В завершение добавьте в класс **WorkerRole** метод, создающий миниатюру для указанного изображения.
(Фрагмент кода – *Introduction to Windows Azure - Ex2 ProcessImage – CS*)
C#

```
public class WorkerRole : RoleEntryPoint
{
...
public void ProcessImage(Stream input, Stream output)
{
int width;
int height;
var originalImage = new Bitmap(input);
if (originalImage.Width > originalImage.Height)
{
width = 128;
height = 128 * originalImage.Height / originalImage.Width;
}
else
{
height = 128;
width = 128 * originalImage.Width / originalImage.Height;
```

```
}
var thumbnailImage = new Bitmap(width, height);

using (Graphics graphics = Graphics.FromImage(thumbnailImage))
{
graphics.InterpolationMode = InterpolationMode.HighQualityBicubic;
graphics.SmoothingMode = SmoothingMode.AntiAlias;
graphics.PixelOffsetMode = PixelOffsetMode.HighQuality;
graphics.DrawImage(originalImage, 0, 0, width, height);
}
thumbnailImage.Save(output, ImageFormat.Jpeg);
}
}
```

12. Прикладная роль также использует сервисы хранения данных Windows Azure, поэтому вам потребуется добавить соответствующие настройки в файл конфигурации, точно так же, как это было сделано в веб-роли. Чтобы добавить настройку, разверните узел **Roles** проекта **GuestBook**, дважды щелкните на **GuestBook_WorkerRole** и перейдите на закладку **Settings**. Нажмите **Add Setting**, дайте настройке имя "*DataConnectionString*", измените **Type** на *ConnectionString*, и нажмите кнопку с многоточием. В диалоге **Storage Account Connection String** выберите вариант **Use the Windows Azure storage emulator** и нажмите **OK**. Нажмите **CTRL + S**, чтобы сохранить изменения.

Практическая работа 3. Развертывание приложения в windows azure
Задача 1 – создание прикладной роли для фоновой обработки данных

Задача 1 – создание сервиса хранения данных и вычислительного сервиса
Развертываемое приложение использует как вычислительные ресурсы, так и хранилище данных. В этой задаче вы создадите учетную запись хранилища, что позволит приложению хранить данные. Дополнительно вам потребуется вычислительный сервис, предназначенный для выполнения кода приложения.
1. Откройте браузер и перейдите на страницу http://windows.azure.com. Введите реквизиты учетной записи Windows Live ID, связанной с учетной записью Windows Azure.

2. Создайте учетную запись хранилища данных. Нажмите кнопку **New Storage Account** на панели инструментов.

3. В диалоге **Create a New Storage Account** выберите подписку в раскрывающемся списке **Choose a subscription**.

4. В текстовом поле **Enter a URL** введите имя создаваемого сервиса, например ***<yourname>guestbook***, где *<yourname>* - уникальное имя. Windows Azure использует это значение для генерации URL конечных точек сервиса.

5. Выберите вариант **Create or choose an affinity group**, затем из раскрывающегося списка значение **Create a new affinity group**.

6. В диалоге **Create a New Affinity Group** введите в поле **Affinity Group Name** имя "guestbook", выберите местоположение (**Location**) и нажмите кнопку **OK**.

7. Вернувшись в диалог **Create a New Storage Account**, нажмите кнопку **Create** для создания хранилища. Дождитесь завершения процесса создания и обновления списка **Storage Accounts**.

Убедитесь в том, что панель **Properties** отображает соответствующие каждому из сервисов **URL**. Запишите имя учетной записи – первый фрагмент назначенного конечным точкам адреса.

8. Теперь нажмите кнопку **View** напротив поля **Primary access key** на панели **Properties**. В диалоге **View Storage Access Keys**, нажмите кнопку **Copy to Clipboard** напротив поля **Primary Access Key**. Это значение потребуется позже для конфигурирования приложения.

9. Теперь создайте вычислительный сервис для запуска исполняемого кода приложения. Выберите раздел **Hosted Services** в левой панели. Нажмите кнопку **New Hosted Service** на панели инструментов.

10. В диалоге **Create a new Hosted Service** выберите из раскрывающегося списка **Choose a subscription** необходимую подписку.

11. Введите имя сервиса в текстовое поле **Enter a name for your service** и выберите для него URL, введя его префикс в текстовое поле **Enter a URL prefix for your service**, например, ***<yourname>guestbook***, где *<yourname>* - уникальное имя. Windows Azure использует это значение для генерации URL конечных точек сервиса.

12. Укажите вариант **Create or choose an affinity group** и выберите из раскрывающегося списка созданную ранее группу **guestbook**.
13. Выберите вариант **Do not Deploy**.
14. Нажмите кнопку **OK**, чтобы создать сервис и дождитесь завершения операции.

15. Не закрывайте окно браузера. Портал потребуется вам для выполнения следующей задачи.

Задача 2 – развертывание приложения с помощью портала Windows Azure

Существует несколько способов развернуть приложение в Windows Azure. Так, набор средств Windows Azure Tools for Visual Studio позволяет создавать и развертывать пакеты непосредственно из Visual Studio. Другой вариант развертывания связан с использованием сценариев Windows Azure Service Management PowerShell Cmdlets и подходит для автоматизации данной процедуры. Наконец, административный портал Windows Azure предоставляет веб-интерфейс, позволяющий выполнить развертывание с помощью браузера. В этой задаче вы развернете приложение в тестовой (staging) среде с использованием портала, но сначала потребуется сгенерировать пакет средствами Visual Studio.

1. Запустите Visual Studio от имени администратора, если это не было сделано ранее: выберите пункт меню **Start | All Programs | Microsoft Visual Studio 2010**. Щелкните правой кнопкой на ярлыке **Microsoft Visual Studio 2010** и выберите элемент **Run as administrator**.
2. В случае появления диалога **User Account Control** нажмите кнопку **Continue**.

3. В меню **File** выберите **Open | Project/Solution**. В диалоге **Open Project** перейдите в подкаталог **Source\Ex3-WindowsAzureDeployment\Begin** в папке с материалами к работе и укажите файл **Begin.sln** в соответствующем предпочитаемому языку каталоге (Visual C# или Visual Basic), затем нажмите **Open**. Вы также можете продолжить работу с решением, полученным после завершения предыдущего шага.
4. Чтобы настроить хранилище, откройте находящийся в проекте **GuestBook** файл **ServiceConfiguration.cscfg**. Замените символы [YOUR_ACCOUNT_NAME] именем учетной записи хранилища, которую вы создали в задаче 1. Если вы следовали рекомендации, то имя записано в формате ***<yourname>guestbook***, где *<yourname>* - уникальное имя. Выполните замену в двух местах – в строке подключения *DataConnectionString* и в строке *Microsoft.WindowsAzure.Plugins.Diagnostics.ConnectionString*.

5. Теперь замените символы [YOUR_ACCOUNT_KEY] значением ключа доступа **Primary Access Key**, также созданным в задаче 1. Аналогично предыдущему пункту, выполните замену в двух местах.

6. Сформируйте пакет для развертывания. Для этого нажмите правой кнопкой на проекте **GuestBook** и выберите пункт **Publish**. В диалоге **Deploy Windows Azure project** выберите вариант **Create Service Package Only** и нажмите кнопку **OK**.
После завершения построения проекта и генерации пакета откроется окно Windows Explorer, отображающего содержимое каталога с собранным пакетом.

7. Переключитесь обратно в административный портал.

8. Выберите созданный на предыдущем шаге вычислительный сервис и нажмите кнопку **New Staging Deployment** на панели инструментов.

9. В диалоге **Create a new Deployment** для выбора местоположения пакета (**Package location**) нажмите на кнопку **Browse Locally**, перейдите в каталог с пакетом и выберите **GuestBook.cspkg**.
10. Теперь выберите расположение файла конфигурации (поле **Configuration File**) – нажмите кнопку **Browse Locally** и выберите файл **ServiceConfiguration.cscfg** из того же каталога, что и в предыдущем шаге.

11. Укажите название развертывания (поле **Deployment name**) – строку, позволяющую идентифицировать его, например, **v1.0**.

12. Нажмите кнопку **OK** для начала развертывания. Убедитесь в том, что портал отобразил диалог с предупреждением. Нажмите **See more details** для просмотра сообщения.

```
<ServiceConfiguration serviceName="GuestBook" xmlns="http://schemas.microsoft.c
 <Role name="GuestBook_WebRole">
  <Instances count="1" />
  <ConfigurationSettings>
   <Setting name="Microsoft.WindowsAzure.Plugins.Diagnostics.ConnectionString"
            value="DefaultEndpointsProtocol=https;
                   AccountName=[YOUR_ACCOUNT_NAME];
                   AccountKey=[YOUR_ACCOUNT_KEY]" />
   <Setting name="DataConnectionString"
            value="DefaultEndpointsProtocol=https;
                   AccountName=[YOUR_ACCOUNT_NAME];
                   AccountKey=[YOUR_ACCOUNT_KEY]" />
 </ConfigurationSettings>
 </Role>
  <Role name="GuestBook_WorkerRole">
   <Instances count="1" />
   <ConfigurationSettings>
    <Setting name="Microsoft.WindowsAzure.Plugins.Diagnostics.ConnectionString"
             value="DefaultEndpointsProtocol=https;
                    AccountName=[YOUR_ACCOUNT_NAME];
                    AccountKey=[YOUR_ACCOUNT_KEY]" />
    <Setting name="DataConnectionString"
             value="DefaultEndpointsProtocol=https;
                    AccountName=[YOUR_ACCOUNT_NAME];
                    AccountKey=[YOUR_ACCOUNT_KEY]" />
  </ConfigurationSettings>
 </Role>
</ServiceConfiguration>
```

13. Нажмите **Yes** чтобы продолжить развертывание. Убедитесь в том, что пакет начал загружаться, для чего на портале отобразился индикатор состояния.

14. Дождитесь завершения развертывания, что может занять несколько минут. Служба развернута и находится в состоянии **Ready**. Убедитесь в том, что портал назначил службе **DNS**, представленный в форме уникального идентификатора. Если коротко – это тестовый URL для доступа к службе.

Задача 3 – изменение числа экземпляров роли

Перед началом тестирования приложения вы сконфигурируете его.

1. В разделе **Hosted Services** выберите сервис GuestBook и нажмите кнопку **Configure**.

2. В диалоге **Configure Deployment** выберите вариант **Edit current configuration**, найдите внутри конфигурации роли **GuestBook_WebRole** элемент **Instances** и измените значение атрибута **count** на *2*. Выполните такую же операцию для прикладной роли **GuestBook_WorkerRole**.

3. Нажмите кнопку **OK** для обновления настроек и дождитесь их вступления в силу.

Задача 4 – проверка приложения в тестовой среде

В этой задаче вы запустите приложение в тестовой среде и проверите его работоспособность.

1. В разделе **Hosted Services** выберите сервис GuestBook и нажмите на ссылке **DNS name** в правой панели**.**

2. Если хотите, вы можете протестировать приложение и добавить запись в гостевую книгу.

Задача 5 – передача приложения в промышленную среду

Теперь, когда вы убедились в работоспособности сервиса, вы можете передать его в промышленную эксплуатацию. Этот процесс связан с реконфигурированием балансировщиков нагрузки, после чего приложение становится доступным по «реальному» адресу.

1. В разделе **Hosted Services** выберите сервис GuestBook и нажмите кнопку **Swap VIP** на панели инструментов.

2. В диалоге **Swap VIPs** нажмите кнопку **OK** для замены тестовой и промышленной сред.

81 Microsoft Tech·Ed Russia 2011. Практическая работа. Основы разработки под Windows Azure.

3. Дождитесь завершения процесса. Обычно на это требуется несколько секунд.

4. Нажмите ссылку **DNS name** для открытия браузера и убедитесь в том, что адрес соответствует промышленному слоту.

5. Даже когда сервис находится в остановленном состоянии (suspended), он потребляет ресурсы, а вы платите за них. После завершения тестирования не забывайте удалить неиспользуемые ресурсы. Чтобы удалить сервис, перейдите в раздел Hosted Services, выберите тип развертывания (staging или production)и остановите его, нажав кнопку Stop на панели инструментов. После останова сервиса нажмите кнопку Delete.

9.2 Методические рекомендации по подготовке письменных работ

Порядок составления и оформления отчета о практической работе

В значительной мере эффективность решения задачи по выполнению практической работы зависит от качества соответствующего отчета. Для этого необходимо соблюдать следующие основные требования по составлению и оформлению отчета, обусловленные соответствующими нормативными документами. Текст отчета должен быть лаконичным и вместе с тем информативным. Текст должен быть изложен с соблюдением правил грамматики. Отчет составляется с обязательным составлением следующих разделов:

1. Заголовок отчета.
2. Цели работы.
3. Методика работы.
4. Порядок выполнения работы (этапы работы).
5. Выводы по работе.

1. В **заголовке отчета** приводятся наименования идентифицирующих признаков: **Отчет о практической работе № 2** по теме, например, «***Фоновая обработка данных с использованием прикладной роли и очереди***», ниже указываются данные студента (фамилия и инициалы, вид обучения, специальность, курс, группа).

2. В разделе **Цель работы** формулируется цели работы студента в соответствии с содержанием раздела «Постановка задачи» данной работы и индивидуального задания студенту на работу.

3. В разделе **Методика работы** указывается методика работы в соответствии с имеющейся формулировкой в разделе «Методика работы» данной работы и при необходимости уточняется в зависимости от содержания конкретного варианта задания студенту на практическую работу.

4. **Порядок выполнения работы**. Приводятся номера и наименования этапов работы, предусмотренные для работы данного Практикума. По каждому из этапов приводится описание выполненных студентом работ, направленных на достижение цели работы. Пропуск какого-либо

из этапов работы Практикума не допускается. В рамках этапов помещается соответствующий иллюстративный материал - таблицы, рисунки (графики), полученные по ходу решения задачи работы. Обозначение иллюстративного материала выполняется в соответствии с правилами, принятыми для публикаций. Обозначение каждой таблицы и рисунка должно иметь номер и наименование. Внутри каждого отчета таблицы и рисунки обозначаются соответственно сквозными номерами. Обозначение таблицы указывается над таблицей, а обозначение рисунка под рисунком. Приводимые в тексте данной работы примеры включать в отчет не разрешается. Применяется только материал, полученный в ходе работы студентом по соответствующему заданию, полученному от преподавателя.

5. Последним разделом отчета являются **выводы** по работе. Это самая сложная и трудная часть работы. Очень важно, чтобы выводы отражали методику, технологию, применяемые программно-аппаратные средства решения задачи. Полезно каждому из этапов работы формулировать не менее одного вывода. Вывод может содержать от одного до трех предложений. Формулировки выводов должны быть конкретными, информативными, лаконичными, по возможности подкрепляться количественными данными.

Оформление отчета выполняется с учетом общепринятых правил. Графическая часть отчетов должна соответствовать правилам графического оформления. Текст отчета набирается в редакторе Word через 1,5 интервала, 14 кегль. Следует использовать шрифт Times New Roman. Заголовки разделов и подразделов выделяются жирным шрифтом. После окончания оформления отчета он проверяется студентом на предмет качество содержания и формы. При условии обнаружения ошибок последние исправляются. После устранения дефектов отчета его экранная форма, или принтерная распечатка предъявляется преподавателю. При условии обнаружения преподавателем ошибок в отчете студент их исправляет и предъявляет отчет преподавателю повторно. Если ошибок нет, то отчет принимается и сохраняется на жестком диске.

Отчет по работе сохраняется студентом в виде отдельного файла. В имени файла указывается фамилия студента и номер выполненной работы. Файл сохраняется в папке с фамилией студента в папке соответствующей студенческой группы. Папка группы создается на первом занятии. В имени папки группы должен присутствовать индекс группы. Папка группы включается в папку «Мои документы».

Приложение 1. Аннотация
рабочей программы дисциплины

АННОТАЦИЯ РАБОЧЕЙ ПРОГРАММЫ ДИСЦИПЛИНЫ

Цель дисциплины: теоретическое и практическое освоение основ облачных вычислений: понятие облака, программное обеспечение и аппаратные средства облачных вычислений, архитектуры облачных приложений и модели облачных инфраструктур, мобильные приложения, основы облачной обработки данных, подготовка к переходу на облачные вычисления, обеспечение безопасности данных в облаке, масштабирование облачной инфраструктуры.

Задачи:

- освоить общие принципы работы с облачной инфраструктурой и приложениями;
- раскрыть особенности использования различных моделей развертывания облачных инфраструктур;
- ознакомиться с архитектурами облачных приложений: технология Grid Computing, транзакционные вычисления;
- ознакомиться с принципами облачной обработки данных на базе решений различных фирм;
- познакомиться с процессом подготовки к переходу на облачные вычисления;
- раскрыть особенности, связанные с обеспечением данных в облаке.

Знать: преимущества облачной инфраструктуры; отличие различных моделей развертывания облачных инфраструктур; принципы облачной обработки данных; принципы создания мобильных приложений для работы в облаке; структуру процесса перехода на облачные вычисления; способы обеспечения защиты информации в облаке.

Уметь: работать с различными облачными сервисами как единолично, так и в команде.

Владеть: навыками работы с комплектами средств разработки мобильных облачных приложений.