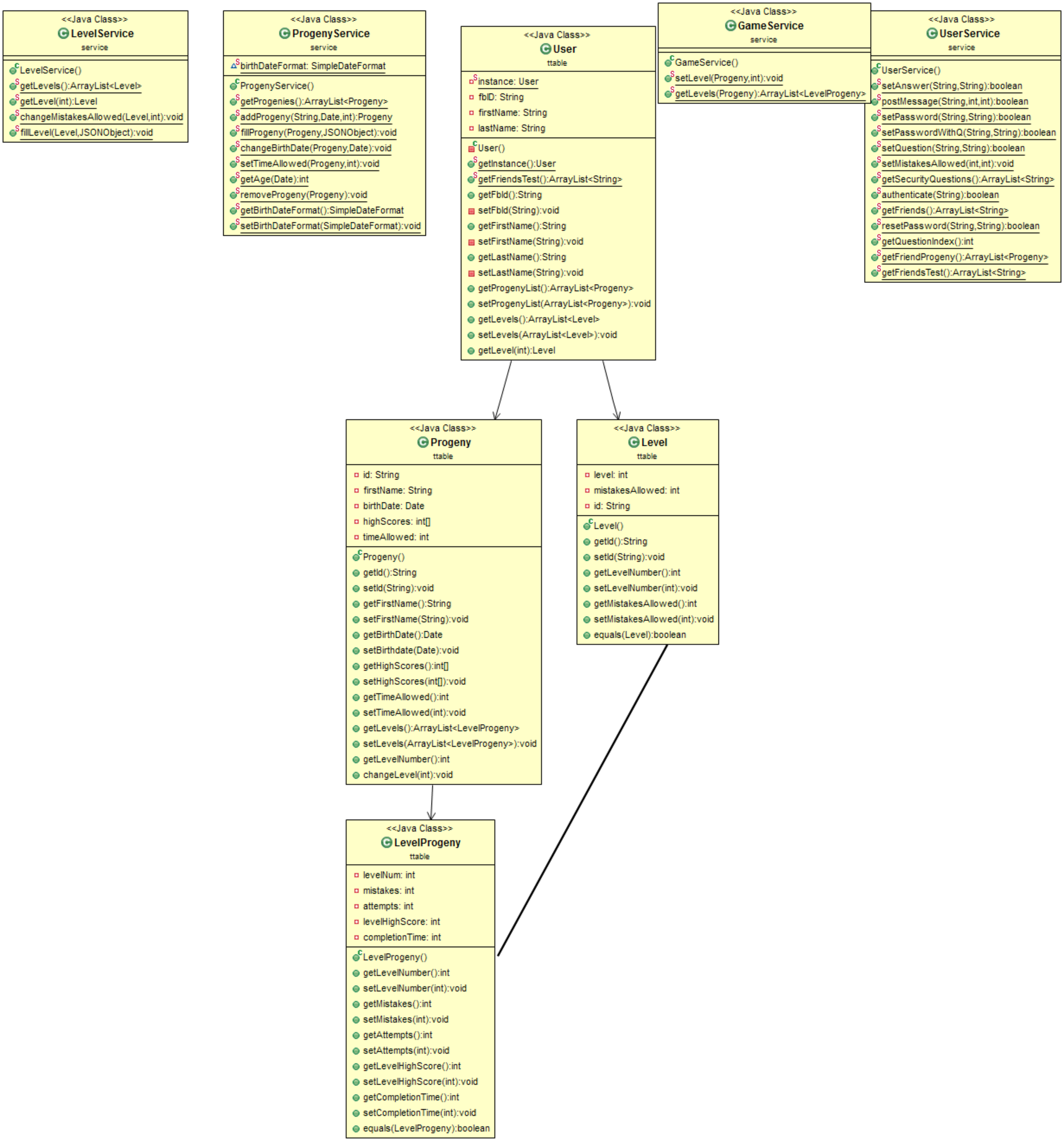
CS 2212a, Section001
Group Assignment 3
Due Date: Friday 22nd November 2013
Group 4
Group Members:
Yaqzan Ali
James Anderson, BSc
James Baron
Taylor Calder, BSc
Chuhan Qin


<<Java Class>>
LevelService
service
LevelService()
getLevels():ArrayList<Level>
getLevel(int):Level
changeMistakesAllowed(Level,int):void
fillLevel(Level,JSONObject):void
<<Java Class>>
ProgenyService
service
birthDateFormat: SimpleDateFormat
ProgenyService()
getProgenies():ArrayList<Progeny>
addProgeny(String,Date,int):Progeny
fillProgeny(Progeny,JSONObject):void
changeBirthDate(Progeny,Date):void
setTimeAllowed(Progeny,int):void
getAge(Date):int
removeProgeny(Progeny):void
getBirthDateFormat():SimpleDateFormat
setBirthDateFormat(SimpleDateFormat):void
<<Java Class>>
User
ttable
instance: User
fbID: String
firstName: String
lastName: String
User()
getInstance():User
getFriendsTest():ArrayList<String>
getFbId():String
setFbId(String):void
getFirstName():String
setFirstName(String):void
getLastName():String
setLastName(String):void
getProgenyList():ArrayList<Progeny>
setProgenyList(ArrayList<Progeny>):void
getLevels():ArrayList<Level>
setLevels(ArrayList<Level>):void
getLevel(int):Level
<<Java Class>>
GameService
service
GameService()
setLevel(Progeny,int):void
getLevels(Progeny):ArrayList<LevelProgeny>
<<Java Class>>
UserService
service
UserService()
setAnswer(String,String):boolean
postMessage(String,int,int):boolean
setPassword(String,String):boolean
setPasswordWithQ(String,String):boolean
setQuestion(String,String):boolean
setMistakesAllowed(int,int):void
getSecurityQuestions():ArrayList<String>
authenticate(String):boolean
getFriends():ArrayList<String>
resetPassword(String,String):boolean
getQuestionIndex():int
getFriendProgeny():ArrayList<Progeny>
getFriendsTest():ArrayList<String>
<<Java Class>>
Progeny
ttable
id: String
firstName: String
birthDate: Date
highScores: int[]
timeAllowed: int
Progeny()
getId():String
setId(String):void
getFirstName():String
setFirstName(String):void
getBirthDate():Date
setBirthdate(Date):void
getHighScores():int[]
setHighScores(int[]):void
getTimeAllowed():int
setTimeAllowed(int):void
getLevels():ArrayList<LevelProgeny>
setLevels(ArrayList<LevelProgeny>):void
getLevelNumber():int
changeLevel(int):void
<<Java Class>>
Level
ttable
level: int
mistakesAllowed: int
id: String
Level()
getId():String
setId(String):void
getLevelNumber():int
setLevelNumber(int):void
getMistakesAllowed():int
setMistakesAllowed(int):void
equals(Level):boolean
<<Java Class>>
LevelProgeny
ttable
levelNum: int
mistakes: int
attempts: int
levelHighScore: int
completionTime: int
LevelProgeny()
getLevelNumber():int
setLevelNumber(int):void
getMistakes():int
setMistakes(int):void
getAttempts():int
setAttempts(int):void
getLevelHighScore():int
setLevelHighScore(int):void
getCompletionTime():int
setCompletionTime(int):void
equals(LevelProgeny):boolean

Testing Plan

Logging on for the First Time

ID: 1
TESTING: Logging on for the first time, new password is too long.
REFERENCES: Requirement 4.c.1
TEST CASE:
1. At start screen, click Settings.
2. Type “0000000” in any of the password fields.
EXPECTED OUTPUT:
The UI will not let you enter a password longer than 6 characters.

ID: 2
TESTING: Logging on for the first time, new password too short.
REFERENCES: Requirement 4.c.1
TEST CASE:
1. At start screen, click Settings.
2. Type in “00” under new password.
3. Type in “00” under retype password.
4. Click Update.
EXPECTED OUTPUT:
Both the new password and retype password fields will turn pink to indicate that your passwords must be at least 3 characters long.

ID: 3
TESTING: Logging on for the first time, new passwords don't match.
REFERENCES: Requirement 4.c.1
TEST CASE:
1. At start screen, click Settings.
2. Type “cs2212” as the old password.
3. Type “cs2211” under new password.
4. Type “cs2210” under retype password.
5. Click Update.
EXPECTED OUTPUT:
The new and retype password fields will turn red, indicating that the passwords do not match.

ID: 4
TESTING: Logging on for the first time with an invalid security answer
REFERENCES: Requirement 4.c.1.1
TEST CASE:
1. At start screen, click Settings.
2. Type in “cs2212” as the password.
3. Select any security question.
4. Leave “-- ANSWER--” in the answer field.
5. Click Update.
EXPECTED OUTPUT:
The answer field turns pink to indicate your security answer is invalid.

ID: 5
TESTING: Logging on for the first time with a blank answer.
REFERENCES: Requirement 4.c.1.1
TEST CASE:
1. At start screen, click settings
2. Type in “cs2212” as the password.
3. Select a security question.
4. Click on the answer field and backspace to remove all text.
5. Click Update.
EXPECTED OUTPUT:
The text field turns pink to indicate your empty security answer is invalid.

ID: 6
TESTING: Logging on for the first time, security answer too long.
REFERENCES: Requirement 4.c.1.1
TEST CASE:
1. At start screen, click Settings.
2. Attempt to type an answer longer than 30 characters.
EXPECTED OUTPUT:
The text field will not let you enter any more characters.

ID: 7
TESTING: Logging on for the first time, valid security question and password.
REFERENCES: Requirement 4.c.1
TEST CASE:
1. At start screen, click Settings.
2. Type in “cs2212” under old password.
3. Type in “cs2212” under new password.
4. Type in “cs2212” under new password.
5. Select a security answer.
6. Type “cs2212” as the security answer.
7. Click Update.
EXPECTED OUTPUT:
You will successfully log on and be sent to a screen where you may add children.

Adding a Child

ID: 8
TESTING: Adding a child, invalid information.
REFERENCES: Requirement 4.c.2
TEST CASE:
1. At start screen, click Settings.
2. Type in the password (default “cs2212”).
3. Click Add (leaving the child information blank).
4. Click Update.
5. Click the Settings button again.
6. Type in the password (default”"cs2212”) and click OK.
EXPECTED OUTPUT:
The blank child will not be in your table of children

ID: 9
TESTING: Adding a child, valid information
REFERENCES: Requirement 4.c.2
TEST CASE:
1. At start screen, click on Settings in the top right.
2. Type in the password (default “cs2212”).
3. Click on the empty field under Add Child: and type “Taylor”.
4. Click on the drop down bar to the right and select “2010”.
5. Click on the drop down menu to the right of that and select “January”.
6. Click on the rightmost drop down menu and select “10”.
7. Click Add.
8. Click the Settings button again.
9. Type in the password (default “cs2212”) and click OK.
EXPECTED OUTPUT:
You should now see a child named Taylor with the specified information in the list of children.

Playing a Level Game

ID: 10
TESTING: Child attempting to start a drill game above their level.
REFERENCES: Requirement 4.d.1
TEST CASE:
1. At start screen, select a child and click OK.
2. Click Drill Menu.
3. Click on a drill game above that child's level.
EXPECTED OUTPUT:
No output, the child will not be able to select the game.

ID: 11
TESTING: Child attempting to start a drill game at or below their level.
REFERENCES: Requirements 4.d.2, 4.d.3
TEST CASE
1. At start screen, select a child and click OK.
2. Click Drill Game.
3. Click on a level at or a below the child's level.
EXPECTED OUTPUT:
A drill game will begin and the child will be able to answer questions.

ID: 12
TESTING: Child finishes a drill game.
REFERENCES: Requirements 4.d.4, 4.d.5
TEST CASE:
1. At start screen, select a child and click OK.
2. Click Drill Game and start a game.
3. Play the game to completion.
EXPECTED OUTPUT:
A congratulations message will appear, and the child will be sent to a new score that will show their score and allow them to post their progress to Facebook or to go on to play the end of level game.

LevelService.java

```
package service;

import java.text.ParseException;
import java.util.ArrayList;
import java.util.Iterator;

import org.json.simple.JSONArray;
import org.json.simple.JSONObject;

import json.JSONFailureException;
import json.Json;
import ttable.Level;
import ttable.Progeny;

public class LevelService {

	/**
	 * Gets an array of the levels (ie. global level settings).
	 *
	 * @return                                  an array of the levels
	 * @throws JSONFailureException  the JSON failure exception
	 */
	public static ArrayList<Level> getLevels() throws JSONFailureException {

		ArrayList<Level> levels = new ArrayList<Level>();

		Json json = new Json();
		JSONObject jsonObj = json.sendRequest("https://jbaron6.cs2212.ca/getlevels");

		JSONArray levelsArray = (JSONArray) jsonObj.get("levels");

		Iterator<?> levelsIt = levelsArray.iterator();

		while(levelsIt.hasNext())
		{
			JSONObject levelObj = (JSONObject)levelsIt.next();

			Level level = new Level();
			fillLevel(level, levelObj);
			levels.add(level);
		}

		return levels;
	}

	public static Level getLevel(int level_number) throws JSONFailureException
	{
		Json json = new Json();
		JSONObject jsonObj = json.sendRequest("https://jbaron6.cs2212.ca/getlevel?
level=" + level_number);

		Level level = new Level();
		fillLevel(level, (JSONObject)jsonObj.get("level"));

		return level;
```

LevelService.java

```
	}

	public static void changeMistakesAllowed(Level level, int mistakes_allowed) throws
JSONFailureException
	{
		Json json = new Json();
		JSONObject jsonObj =
json.sendRequest("https://jbaron6.cs2212.ca/changelevelmistakesallowed?level=" +
level.getLevelNumber() + "&mistakes_allowed=" + mistakes_allowed);

		fillLevel(level, (JSONObject)jsonObj.get("level"));
	}

	public static void fillLevel(Level level, JSONObject level_data)
	{


level.setMistakesAllowed(Integer.parseInt(level_data.get("mistakes_allowed").toString()));
		level.setLevelNumber(Integer.parseInt(level_data.get("level").toString()));
		level.setId(level_data.get("id").toString());
	}


}
```

LevelServiceTest.java

```
package service;

import static org.junit.Assert.*;

import java.util.ArrayList;

import json.JSONFailureException;

import org.junit.AfterClass;
import org.junit.BeforeClass;
import org.junit.Test;

import ttable.Level;

public class LevelServiceTest {

	@BeforeClass
	public static void setUpBeforeClass() throws Exception {
	}

	@AfterClass
	public static void tearDownAfterClass() throws Exception {
	}

	@Test
	public void testChangeMistakesAllowed()
	{
		Level level = null;

		try {
			level = LevelService.getLevel(4);
		} catch (JSONFailureException e) {
			fail(e.getMessage());
		}

		int mistakes_allowed = level.getMistakesAllowed();
		int new_mistakes_allowed = 700;

		if (mistakes_allowed == 700)
			new_mistakes_allowed = 30;

		try {
			LevelService.changeMistakesAllowed(level, new_mistakes_allowed);
		} catch (JSONFailureException e) {
			fail(e.getMessage());
		}

		assertEquals(new_mistakes_allowed, level.getMistakesAllowed());

		try {
			LevelService.changeMistakesAllowed(level, mistakes_allowed);
		} catch (JSONFailureException e) {
			fail(e.getMessage());
		}
```

LevelServiceTest.java

```
			assertEquals(mistakes_allowed, level.getMistakesAllowed());

		}

		@Test
		public void testGetLevels() {

			ArrayList<Level> levels = null;

			try {
				levels = LevelService.getLevels();
			} catch (JSONFailureException e) {
				fail(e.getMessage());
			}

			assertEquals(12, levels.size());


			assertEquals(4, levels.get(3).getLevelNumber());
			assertEquals(7, levels.get(6).getLevelNumber());
			assertEquals(12, levels.get(11).getLevelNumber());
		}

		@Test
		public void testGetLevel()
		{
			Level level4 = null;
			Level level7 = null;
			Level level12 = null;

			try {
				level4 = LevelService.getLevel(4);
				level7 = LevelService.getLevel(7);
				level12 = LevelService.getLevel(12);
			} catch (JSONFailureException e) {
				fail(e.getMessage());
			}

			assertEquals(4, level4.getLevelNumber());
			assertEquals(7, level7.getLevelNumber());
			assertEquals(12, level12.getLevelNumber());
		}

}
```

ProgenyService.java

```
ackage service;

import java.util.Date;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.GregorianCalendar;
import java.util.Iterator;

import org.json.simple.JSONArray;
import org.json.simple.JSONObject;

import json.JSONFailureException;
import json.Json;
import ttable.Progeny;

/**
 * The Class ProgenyService.
 *
 * @author James Baron
 * @author James Anderson
 */
public class ProgenyService {

	static SimpleDateFormat birthDateFormat = new SimpleDateFormat("yyyy-MM-dd");


	/**
	 * Gets an array of progenies.
	 *
	 * @return                                    an array of progeny
	 * @throws JSONFailureException the jSON failure exception
	 */
	public static ArrayList<Progeny> getProgenies() throws JSONFailureException
	{
		ArrayList<Progeny> progenies = new ArrayList<Progeny>();

		Json json = new Json();
		JSONObject jsonObj =
json.sendRequest("https://jbaron6.cs2212.ca/getprogenies");


		JSONArray progeniesArray = (JSONArray) jsonObj.get("progenies");

		Iterator<?> progeniesIt = progeniesArray.iterator();

		while(progeniesIt.hasNext())
		{
			JSONObject progenyObj = (JSONObject)progeniesIt.next();

			Progeny progeny = new Progeny();
			fillProgeny(progeny, progenyObj);
			progenies.add(progeny);
		}
```

ProgenyService.java

```
		return progenies;
	}

	/**
	 * Adds the progeny.
	 *
	 * @param name the name
	 * @param birthDate the birth date
	 * @return the progeny
	 * @throws JSONFailureException the jSON failure exception
	 */
	public static Progeny addProgeny(String name, Date birthDate, int timeAllowed)
throws JSONFailureException
	{
		Json json = new Json();
		JSONObject progeny_obj =
json.sendRequest("https://jbaron6.cs2212.ca/addprogeny?first_name=" + name +
		"&birth_date=" + birthDateFormat.format(birthDate) +
	"&time_allowed=" + timeAllowed
				);

		Progeny progeny = new Progeny();

		fillProgeny(progeny, (JSONObject)progeny_obj.get("progeny"));

		return progeny;
	}

	public static void fillProgeny(Progeny progeny, JSONObject progeny_data)
	{
		progeny.setFirstName((String)progeny_data.get("first_name"));

		try {

	progeny.setBirthdate(birthDateFormat.parse((String)progeny_data.get("birth_date")));
		} catch (ParseException e) {}



progeny.setTimeAllowed(Integer.parseInt(progeny_data.get("time_allowed").toString()));

		progeny.setId(progeny_data.get("id").toString());
	}

	public static void changeBirthDate(Progeny progeny, Date birthDate) throws
JSONFailureException
	{
		SimpleDateFormat birth_date_format = new SimpleDateFormat("yyyy-MM-dd");

		Json json = new Json();
		JSONObject progeny_data =
json.sendRequest("https://jbaron6.cs2212.ca/changeprogenybirthdate?progeny_id=" +
progeny.getId() + "&birth_date=" + birth_date_format.format(birthDate));
```

ProgenyService.java

```
			fillProgeny(progeny, (JSONObject)progeny_data.get("progeny"));
		}

		/**
		 * Sets the specified progeny's time allowed per level.
		 *
		 * @param progeny
		 * @param timeAllowed
		 * @throws JSONFailureException
		 */
		public static void setTimeAllowed(Progeny progeny, int timeAllowed) throws
JSONFailureException
		{

			Json json = new Json();
			JSONObject progeny_data =
json.sendRequest("https://jbaron6.cs2212.ca/changeprogenytimeallowed?progeny_id=" +
progeny.getId() + "&time_allowed=" + timeAllowed);

			fillProgeny(progeny, (JSONObject)progeny_data.get("progeny"));

		}


		/**
		 * Calculates the child's current age.
		 *
		 * @param birthDate		the child's date of birth
		 * @return				the child's current age.
		 */
		public static int getAge(Date birthDate) {
			GregorianCalendar calPresent = new GregorianCalendar();
			calPresent.setTime(new Date());
			GregorianCalendar calBirth = new GregorianCalendar();
			calBirth.setTime(birthDate);

			int age = calPresent.get(Calendar.YEAR) - calBirth.get(Calendar.YEAR);

			if (calBirth.get(Calendar.MONTH) > calPresent.get(Calendar.MONTH)) {
				age--;
			}
			else if (calBirth.get(Calendar.MONTH) == calPresent.get(Calendar.MONTH)) {
				if (calBirth.get(Calendar.DAY_OF_MONTH) >
calPresent.get(Calendar.DAY_OF_MONTH)) {
					age--;
				}
			}
			return age;
		}

		/**
		 * Removes the specified progeny.
		 *
		 * @param progeny the progeny
		 * @throws JSONFailureException the jSON failure exception
```

ProgenyService.java

```
	 */
	public static void removeProgeny(Progeny progeny) throws JSONFailureException {
		Json json = new Json();
		json.sendRequest("https://jbaron6.cs2212.ca/removeprogeny?progeny_id=" +
progeny.getId());
	}

	/**
	 * Gets the date format used by the server.
	 *
	 * @return		the date format used by the server.
	 */
	public static SimpleDateFormat getBirthDateFormat() {
		return birthDateFormat;
	}

	/**
	 * Sets the date format used by the server.
	 *
	 * @param birthDateFormat		the date format used by the server.
	 */
	public static void setBirthDateFormat(SimpleDateFormat birthDateFormat) {
		ProgenyService.birthDateFormat = birthDateFormat;
	}
}
```

ProgenyServiceTest.java

```
package service;

import static org.junit.Assert.*;

import java.text.ParseException;
import java.util.ArrayList;
import java.util.Date;

import json.JSONFailureException;

import org.json.simple.JSONObject;
import org.json.simple.JSONValue;
import org.junit.AfterClass;
import org.junit.BeforeClass;
import org.junit.Test;

import ttable.Progeny;

public class ProgenyServiceTest {

	@BeforeClass
	public static void setUpBeforeClass() throws Exception {
	}

	@AfterClass
	public static void tearDownAfterClass() throws Exception {
	}

	@Test
	public void testGetProgenies() {

		Date birthDate = null;
		int fiveMinutes = 3000;

		try {
			birthDate = ProgenyService.getBirthDateFormat().parse("1982-05-19");
		} catch (ParseException e1) {
		}

		try {
			ProgenyService.addProgeny("James", birthDate, fiveMinutes);
		} catch (JSONFailureException e) {
			fail(e.getMessages().get(0));
		}

		try {
			ProgenyService.addProgeny("Frank", birthDate, fiveMinutes);
		} catch (JSONFailureException e) {
			fail(e.getMessages().get(0));
		}


		try {
			ProgenyService.addProgeny("Taylor", birthDate, fiveMinutes);
		} catch (JSONFailureException e) {
```

ProgenyServiceTest.java

```
			fail(e.getMessages().get(0));
		}

		ArrayList<Progeny> progenies = null;

		try {
			progenies = ProgenyService.getProgenies();
		} catch (JSONFailureException e) {
			fail(e.getMessages().get(0));
		}

		assertEquals("James", progenies.get(progenies.size() - 3).getFirstName());
		assertEquals("Frank", progenies.get(progenies.size() - 2).getFirstName());
		assertEquals("Taylor", progenies.get(progenies.size() - 1).getFirstName());

		try {
			ProgenyService.removeProgeny(progenies.get(progenies.size() - 3));
			ProgenyService.removeProgeny(progenies.get(progenies.size() - 2));
			ProgenyService.removeProgeny(progenies.get(progenies.size() - 1));
		} catch (JSONFailureException e) {
			fail(e.getMessages().get(0));
		}
	}

	@Test
	public void testAddProgeny() {

		Date birthDate = null;
		int fiveMinutes = 3000;

		try {
			birthDate = ProgenyService.getBirthDateFormat().parse("1982-05-19");
		} catch (ParseException e1) {
		}

		String name = "Jeff";
		Progeny progeny = null;

		try {
			progeny = ProgenyService.addProgeny(name, birthDate, fiveMinutes);
		} catch (JSONFailureException e) {
			fail(e.getMessages().get(0));
		}

		assertTrue(progeny.getBirthDate().equals(birthDate));
		assertEquals(name, progeny.getFirstName());
		assertEquals(fiveMinutes, progeny.getTimeAllowed());

		try {
			ProgenyService.removeProgeny(progeny);
		} catch (JSONFailureException e) {
			fail(e.getMessages().get(0));
		}
	}
```

ProgenyServiceTest.java

```
	@Test
	public void testChangeBirthDate() {
		Date firstBirthDate = null;
		Date secondBirthDate = null;
		int fiveMinutes = 3000;

		try {
			firstBirthDate = ProgenyService.getBirthDateFormat().parse("1982-05-
19");
			secondBirthDate = ProgenyService.getBirthDateFormat().parse("1942-07-
12");
		} catch (ParseException e1) {
		}

		String name = "Bob";
		Progeny progeny = null;

		try {
			progeny = ProgenyService.addProgeny(name, firstBirthDate, fiveMinutes);
		} catch (JSONFailureException e) {
			fail(e.getMessages().get(0));
		}

		try {
			ProgenyService.changeBirthDate(progeny, secondBirthDate);
		} catch (JSONFailureException e1) {
			fail(e1.getMessages().get(0));
		}

		assertTrue(progeny.getBirthDate().equals(secondBirthDate));

		try {
			ProgenyService.removeProgeny(progeny);
		} catch (JSONFailureException e) {
			fail(e.getMessages().get(0));
		}
	}

	@Test
	public void testChangeTimeAllowed()
	{
		Date birthDate = null;
		int fiveMinutes = 3000;
		int sixMinutes = 3600;

		try {
			birthDate = ProgenyService.getBirthDateFormat().parse("1982-05-19");
		} catch (ParseException e1) {
		}

		String name = "Dan";
		Progeny progeny = null;

		try {
			progeny = ProgenyService.addProgeny(name, birthDate, fiveMinutes);
```

ProgenyServiceTest.java

```
		} catch (JSONFailureException e) {
			fail(e.getMessages().get(0));
		}

		assertEquals(fiveMinutes, progeny.getTimeAllowed());

		try {
			ProgenyService.setTimeAllowed(progeny, sixMinutes);
		} catch (JSONFailureException e1) {
			fail(e1.getMessages().get(0));
		}

		assertEquals(sixMinutes, progeny.getTimeAllowed());

		try {
			ProgenyService.removeProgeny(progeny);
		} catch (JSONFailureException e) {
			fail(e.getMessages().get(0));
		}

	}

	@Test
	public void testFillProgeny() {
		JSONObject progeny_data = (JSONObject)
JSONValue.parse("{\"first_name\":\"James\",\"birth_date\":\"1982-05-
19\",\"id\":77,\"time_allowed\":3000}");

		Progeny progeny = new Progeny();

		ProgenyService.fillProgeny(progeny, progeny_data);

		assertEquals("James", progeny.getFirstName());

		try {

	assertTrue(progeny.getBirthDate().equals(ProgenyService.birthDateFormat.parse("1982-
05-19")));
		} catch (ParseException e) {
			fail(e.getMessage());
		}

		assertEquals("77", progeny.getId());
		assertEquals(3000, progeny.getTimeAllowed());
	}



	@Test
	public void testGetAge() {

		int age = 0;

		try {
			age = ProgenyService.getAge(ProgenyService.birthDateFormat.parse("1982-
```

ProgenyServiceTest.java

```
05-19"));
			} catch (ParseException e) {
				fail(e.getMessage());
			}

			assertEquals(31, age);
		}

}
```

UserService.java

```
package service;

import java.util.ArrayList;
import java.util.Iterator;

import org.json.simple.JSONArray;
import org.json.simple.JSONObject;

import ttable.Progeny;
import json.JSONFailureException;
import json.Json;
/**
 *
 * @author James Baron
 * @author James Anderson
 * @version 1.1
 */
public class UserService {

	/**
	 * Sets the answer to the user's password recovery question.
	 *
	 * @param answer the answer
	 * @param password the password
	 * @return true, if successful
	 * @throws JSONFailureException the jSON failure exception
	 */
	public static boolean setAnswer(String answer, String password) throws
JSONFailureException
	{
		Json json = new Json();
		json.sendRequest("https://jbaron6.cs2212.ca/setanswer?answer=" + answer +
"&password=" + password);

		return true;
	}

	/**
	 * Posts a message to Facebook to indicate a child's score
	 *
	 * @param name		the child's name
	 * @param score		the score
	 * @param level		the level
	 * @return true, if successful
	 * @throws JSONFailureException the jSON failure exception
	 */
	public static boolean postMessage(String name, int score, int level) throws
JSONFailureException
	{
		Json json = new Json();
		String message = (name + " just reached a score of " + score + " on level " +
level + "!");
		json.sendRequest("https://jbaron6.cs2212.ca/postmessage?message=" + message);

		return true;
```

UserService.java

```
	}

	/**
	 * Sets the user's password.
	 *
	 * @param oldPassword the old password
	 * @param newpassword the new password
	 * @return true, if successful
	 * @throws JSONFailureException the jSON failure exception
	 */
	public static boolean setPassword(String oldPassword, String newpassword) throws
JSONFailureException
	{
		Json json = new Json();

		json.sendRequest("https://jbaron6.cs2212.ca/setpassword?new_password=" +
newpassword + "&old_password=" + oldPassword);

		return true;
	}

	/**
	 * Sets the user's password using the security question instead of old password.  If
the
	 *  security question is input incorrectly, the password is automatically reset to
the
	 *  default, cs2212.
	 *
	 * @param questionAns                the input answer to the security question.
	 * @param newpassword                the new password to be changed to.
	 * @return                                 true if successful, false otherwise.
	 * @throws JSONFailureException  the JSON exception thrown if the password change is
unsuccessful.
	 */
	public static boolean setPasswordWithQ(String questionAns, String newpassword)
throws JSONFailureException {
		Json json = new Json();

		json.sendRequest("https://jbaron6.cs2212.ca/resetpassword?answer=" +
questionAns + "&new_password=" + newpassword);

		return true;
	}

	/**
	 * Sets the user's password recovery question.
	 *
	 * @param question the question
	 * @param password the password
	 * @return true, if successful
	 * @throws JSONFailureException the jSON failure exception
	 */
	public static boolean setQuestion(String questionNumber, String password) throws
JSONFailureException
	{
```

UserService.java

```
		Json json = new Json();
		json.sendRequest("https://jbaron6.cs2212.ca/setquestion?question=" +
questionNumber + "&password=" + password);

		return true;
	}

	/**
	 * Sets the number of mistakes allowed for a specified level.
	 *
	 * @param mistakesAllowed the new mistakes allowed
	 */
	public static void setMistakesAllowed(int level, int mistakesAllowed) throws
JSONFailureException {
		//TODO:  server call to update
	}

	/**
	 * Gets the current user's password recovery questions.
	 *
	 * @return the questions		the array of security questions
	 * @throws JSONFailureException the JSON failure exception
	 */
	public static ArrayList<String> getSecurityQuestions() throws JSONFailureException
	{
		ArrayList<String> questions = new ArrayList<String>();

		Json json = new Json();
		JSONObject jsonObj =
json.sendRequest("https://jbaron6.cs2212.ca/getquestions");


		JSONArray questionsArray = (JSONArray) jsonObj.get("questions");

		Iterator<?> questionsIt = questionsArray.iterator();

		while(questionsIt.hasNext())
			questions.add((String)questionsIt.next());

		return questions;
	}

	/**
	 * Authenticate the user.
	 *
	 * @param password the password
	 * @return true, if successful
	 * @throws JSONFailureException the jSON failure exception
	 */
	public static boolean authenticate(String password) throws JSONFailureException
	{
		Json json = new Json();

		//Will throw error if success fail.  That error will have messages.
		json.sendRequest("https://jbaron6.cs2212.ca/authenticate?password=" +
```

UserService.java

```
password);

		return true;
	}

	/**
	 * Gets the users friends list
	 *
	 * @return 					an array of friends
	 * @throws JSONFailureException the jSON failure exception
	 */
	public static ArrayList<String> getFriends() throws JSONFailureException
	{
		ArrayList<String> friendList;

		friendList = new ArrayList<String>();

		// TODO: make server call and parse list

		return friendList;
	}



	/**
	 * Reset the user's password.
	 *
	 * @param oldPassword 		the old password
	 * @param newPassword 		the new password
	 * @return 					true, if successful
	 * @throws JSONFailureException the jSON failure exception
	 */
	public static boolean resetPassword(String oldPassword, String newPassword) throws
JSONFailureException
	{
		Json json = new Json();
		json.sendRequest("https://jbaron6.cs2212.ca/resetpassword?answer=" +
oldPassword + "&new_password=" + newPassword);

		return true;
	}

	public static int getQuestionIndex() throws JSONFailureException
	{
		Json json = new Json();
		JSONObject result =
json.sendRequest("https://jbaron6.cs2212.ca/getquestionindex");

		return result.get("question_index") == null ? -1 :
Integer.parseInt((String)result.get("question_index"));
	}

	/**
	 * Gets a friend's progeny.
	 *
```

UserService.java

```
	 * @return                                    an array of progeny
	 * @throws JSONFailureException the jSON failure exception
	 */
	public static ArrayList<Progeny> getFriendProgeny() throws JSONFailureException
	{
		ArrayList<Progeny> progenyList;

		progenyList = new ArrayList<Progeny>();

		// TODO: make server call and parse list

		return progenyList;
	}

	// ** NOTE ** //
		/**
		 * This is a dummy method that emulates how getFriends will work I'm
		 * including it in lieu of just creating a string array in the StatsMenu2
		 * file.
		 *
		 * @return an array of friends
		 * @throws JSONFailureException
		 *             the jSON failure exception
		 */
		public static ArrayList<String> getFriendsTest()
				throws JSONFailureException {
			ArrayList<String> friendList = new ArrayList<String>();

			friendList
					.add("James
Anderson;http://jbaron6.cs2212.ca/img/profilePictures/1.jpg");
			friendList
					.add("Yaqzan
Ali;http://jbaron6.cs2212.ca/img/profilePictures/2.jpg");
			friendList
					.add("Chuhann
Frank;http://jbaron6.cs2212.ca/img/profilePictures/3.jpg");
			friendList
					.add("Taylor
Joseph;http://jbaron6.cs2212.ca/img/profilePictures/4.jpg");
			friendList
					.add("James
Baron;http://jbaron6.cs2212.ca/img/profilePictures/5.jpg");

			// TODO: make server call and parse list

			return friendList;
		}
	}
```

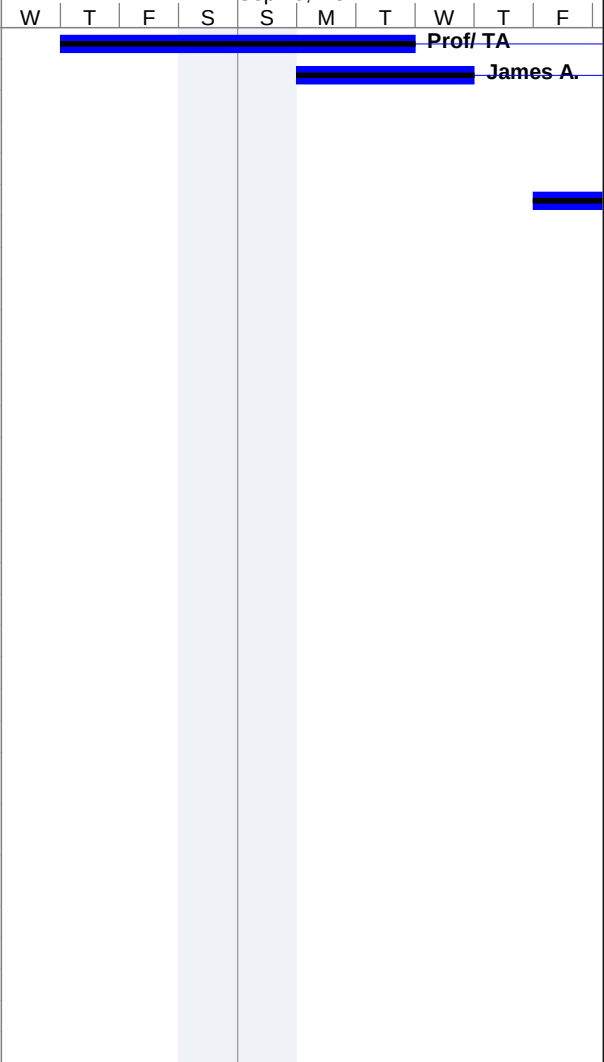
UserServiceTest.java

```
package service;

import static org.junit.Assert.*;
import json.JSONFailureException;

import org.junit.AfterClass;
import org.junit.BeforeClass;
import org.junit.Test;

public class UserServiceTest {

	@BeforeClass
	public static void setUpBeforeClass() throws Exception {
	}

	@AfterClass
	public static void tearDownAfterClass() throws Exception {
	}

	@Test
	public void testSetAnswer() {
		fail("Not yet implemented");
	}

	@Test
	public void testPostMessage() {
		fail("Not yet implemented");
	}

	@Test
	public void testSetPassword() {
		fail("Not yet implemented");
	}

	@Test
	public void testSetPasswordWithQ() {
		fail("Not yet implemented");
	}

	@Test
	public void testSetQuestion() {
		fail("Not yet implemented");
	}

	@Test
	public void testSetMistakesAllowed() {
		fail("Not yet implemented");
	}

	@Test
	public void testGetSecurityQuestions() {
		fail("Not yet implemented");
	}

	@Test
```

UserServiceTest.java

```
	public void testAuthenticate() {
		fail("Not yet implemented");
	}

	@Test
	public void testGetFriends() {
		fail("Not yet implemented");
	}

	@Test
	public void testResetPassword() {
		fail("Not yet implemented");
	}

	@Test
	public void testGetQuestionIndex() {

		int question_index = -2;

		try {
			question_index = UserService.getQuestionIndex();
		} catch (JSONFailureException e) {
			fail(e.getMessages().get(0));
		}

		assertTrue(question_index > -2 && question_index < 3);
	}

	@Test
	public void testGetFriendProgeny() {
		fail("Not yet implemented");
	}

	@Test
	public void testGetFriendsTest() {
		fail("Not yet implemented");
	}

}
```

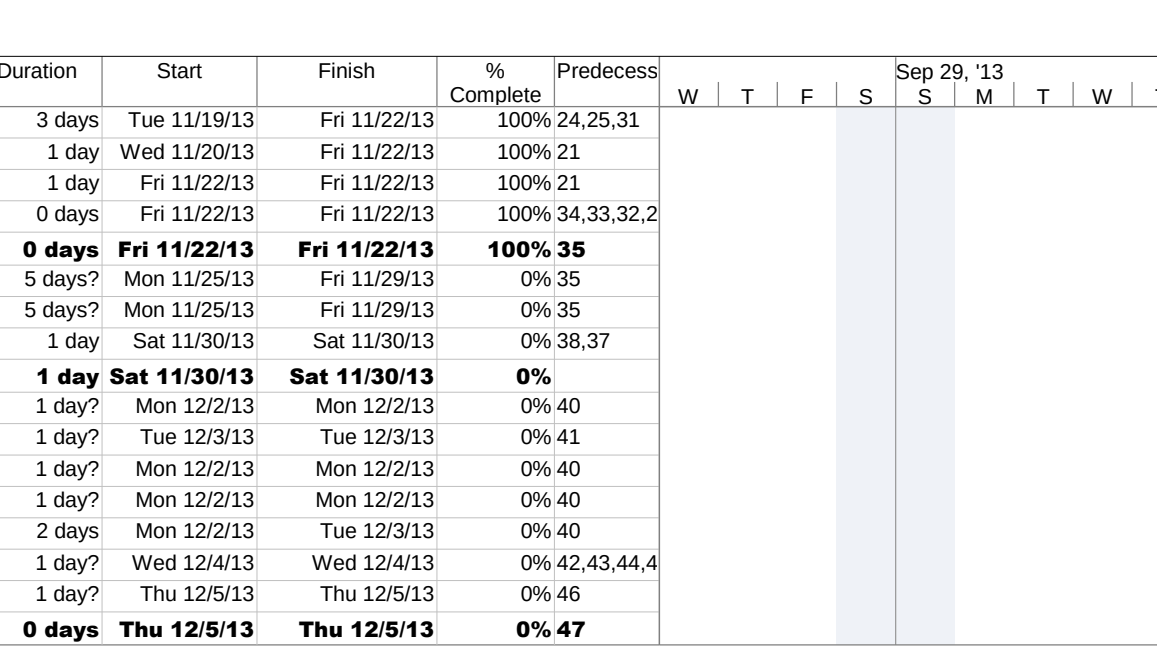
ID	Task Name	Platform	Duration	Start	Finish	% Complete	Predecess
1	Set Up Pivitol Tracker	Pivotal Tracker	4 days	Thu 9/26/13	Tue 10/1/13	100%	
2	Complete Personnel Snapshot	Facebook	3 days	Mon 9/30/13	Wed 10/2/13	100%	
3	Complete User Stories	Pivotal Tracker	6 days	Mon 10/7/13	Mon 10/14/13	100%	
4	Complete UML Use Case Diagrams	UMLet	2.75 days	Fri 10/11/13	Wed 10/16/13	100%	3
5	Input UCD Into Pivitol Tracker	Pivotal Tracker	2 days	Wed 10/16/13	Fri 10/18/13	100%	4,1
6	Learn MS Visio	MS Visio	2 days	Fri 10/4/13	Mon 10/7/13	100%	
7	Complete UML Class Diagrams	UMLet	4 days	Mon 10/7/13	Thu 10/10/13	100%	6
8	Learn MS Project	MS Project	1 day	Tue 10/8/13	Tue 10/8/13	100%	
9	Complete Project Plan	MS Project	4 days	Thu 10/10/13	Tue 10/15/13	100%	8
10	Make Title Page	PDF	1 day	Thu 10/17/13	Thu 10/17/13	100%	
11	**Submit Assignment 1**	**OWL**	**0 days**	**Fri 10/18/13**	**Fri 10/18/13**	**100%**	**2,5,9,10,**
12	Write Classes For Password IO	java	4 days	Fri 10/18/13	Thu 10/24/13	100%	11
13	Set up Git Hub	Eclipse, GitHub	3 days	Tue 10/29/13	Thu 10/31/13	100%	11
14	Revise UML Diagrams	UMLet	2 days	Mon 10/21/13	Tue 10/22/13	100%	11
15	Design Logos and Images for GUI	Photoshop	7 days	Thu 10/24/13	Fri 11/1/13	100%	11
16	Complete User Interface Design	Java	9 days	Fri 10/18/13	Thu 10/31/13	100%	11
17	Complete Class Descriptions	PDF	6 days	Mon 10/21/13	Mon 10/28/13	100%	11
18	Update Project Plan	MS Project	1 day	Mon 10/21/13	Mon 10/21/13	100%	11
19	Complete PDF Report	PDF	1 day	Fri 11/1/13	Mon 11/4/13	100%	14,16,18,1
20	Complete Group4Readme.txt file	TXT	2 days	Sun 11/3/13	Mon 11/4/13	100%	11
21	Complete ZIP File	ZIP	2 days	Sun 11/3/13	Mon 11/4/13	100%	20,19
22	**Submit Assignment 2**	**OWL**	**0 days**	**Mon 11/4/13**	**Mon 11/4/13**	**100%**	**21**
23	Revise UML Diagrams	UMLet	4 days?	Tue 11/5/13	Fri 11/8/13	100%	22
24	Complete Test Plan for Administrative User Creating a New Category	PDF	3 days?	Tue 11/5/13	Thu 11/7/13	100%	22
25	Complete Test Plan for Administrative User Bulkloading a File	PDF	3 days?	Fri 11/8/13	Tue 11/12/13	100%	22
26	Complete Test Plan for a Player Challanging another User	PDF	3 days	Wed 11/13/13	Mon 11/18/13	100%	22
27	Clean up GUI	Java	6 days	Mon 11/11/13	Mon 11/18/13	100%	22
28	Work on Level Game	Java	6 days	Wed 11/13/13	Fri 11/22/13	67%	
29	Work on Persistence Requirements		10 days?	Mon 11/11/13	Fri 11/22/13	90%	
30	Work on Drill Game	Java	10 days	Mon 11/11/13	Fri 11/22/13	80%	
31	Generate pictures for game	Photoshop	6 days	Mon 11/11/13	Mon 11/18/13	100%	



Project: Project1
Date: Fri 11/22/13

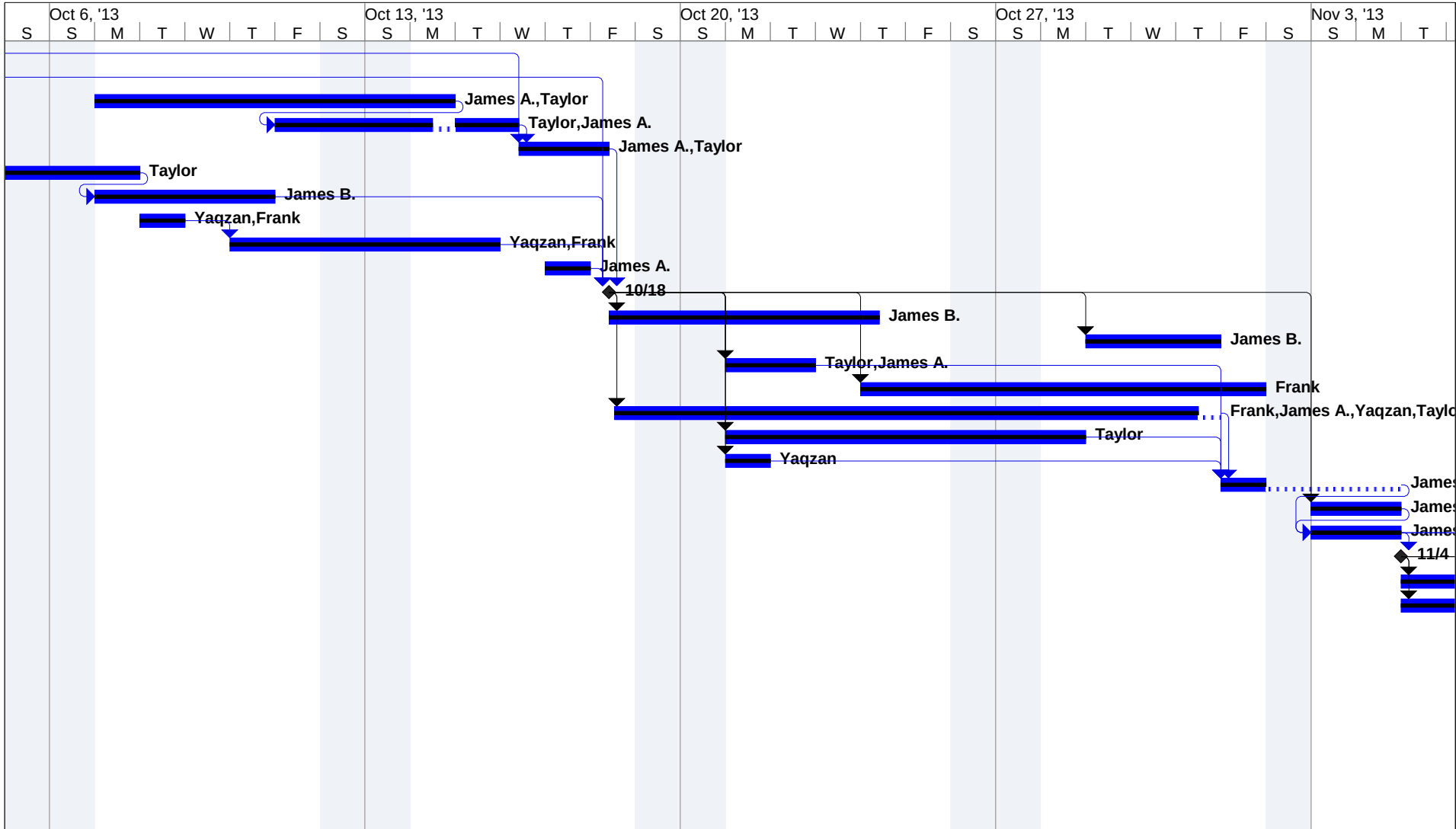
Task	Inactive Milestone	Finish-only	
Split	Inactive Summary	External Tasks	
Milestone	Manual Task	External Milestone	
Summary	Duration-only	Progress	
Project Summary	Manual Summary Rollup	Deadline	
External Tasks	Manual Summary		
External Milestone	Start-only		

ID	Task Name	Platform	Duration	Start	Finish	% Complete	Predecess
32	Complete Automated Junit Test Suit	PDF	3 days	Tue 11/19/13	Fri 11/22/13	100%	24,25,31
33	Update Project Plan	MS Project	1 day	Wed 11/20/13	Fri 11/22/13	100%	21
34	Update Group4Readme.txt	TXT	1 day	Fri 11/22/13	Fri 11/22/13	100%	21
35	Complete Zip File	ZIP	0 days	Fri 11/22/13	Fri 11/22/13	100%	34,33,32,2
36	**Submit Assignment 3**	**OWL**	**0 days**	**Fri 11/22/13**	**Fri 11/22/13**	**100%**	**35**
37	Finish Level Game		5 days?	Mon 11/25/13	Fri 11/29/13	0%	35
38	Finish Final Game		5 days?	Mon 11/25/13	Fri 11/29/13	0%	35
39	Hand In via GitHub	GitHub	1 day	Sat 11/30/13	Sat 11/30/13	0%	38,37
40	**Group Final Code Submission**	**GitHub**	**1 day**	**Sat 11/30/13**	**Sat 11/30/13**	**0%**	
41	Complete Title Page of User Manual	PDF	1 day?	Mon 12/2/13	Mon 12/2/13	0%	40
42	Complete User Manual	PDF	1 day?	Tue 12/3/13	Tue 12/3/13	0%	41
43	Gather Screenshots	PDF	1 day?	Mon 12/2/13	Mon 12/2/13	0%	40
44	Finish Project Plan Document	PDF	1 day?	Mon 12/2/13	Mon 12/2/13	0%	40
45	Finalize Source Code	IDE	2 days	Mon 12/2/13	Tue 12/3/13	0%	40
46	Complete Final Zip File	ZIP	1 day?	Wed 12/4/13	Wed 12/4/13	0%	42,43,44,4
47	Hand In Zip File Via Github	GitHub	1 day?	Thu 12/5/13	Thu 12/5/13	0%	46
48	**Submit Assignment 4**	**OWL&GitHub**	**0 days**	**Thu 12/5/13**	**Thu 12/5/13**	**0%**	**47**



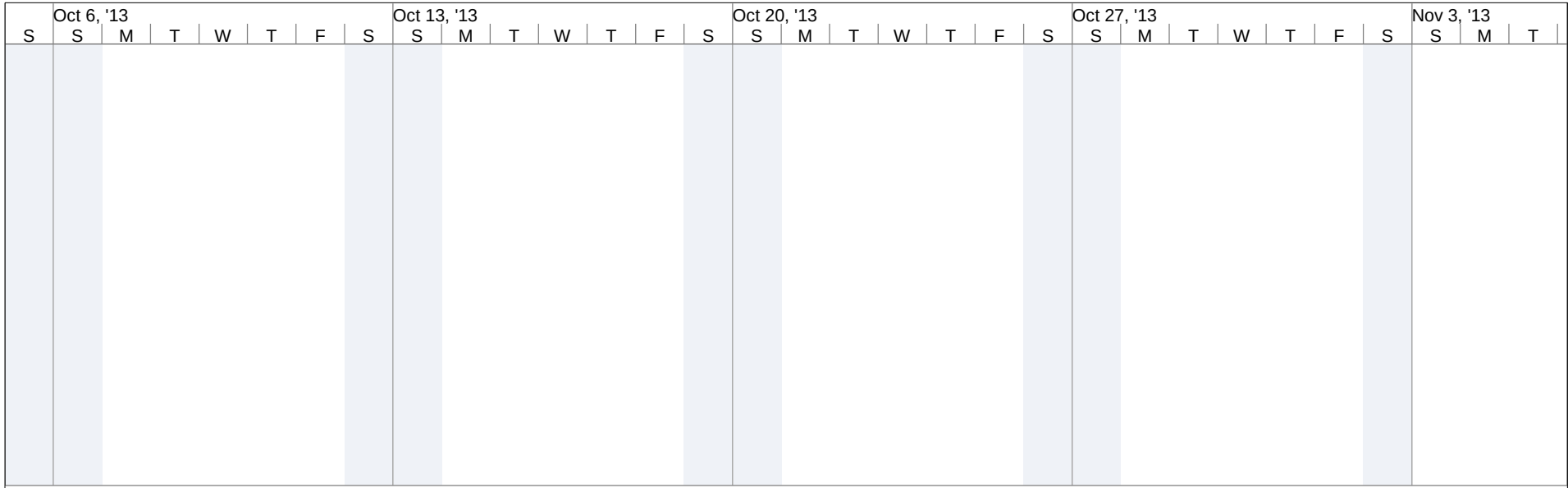
Project: Project1
Date: Fri 11/22/13

Task
Split
Milestone
Summary
Project Summary
External Tasks
External Milestone
Inactive Milestone
Inactive Summary
Manual Task
Duration-only
Manual Summary Rollup
Manual Summary
Start-only
Finish-only
External Tasks
External Milestone
Progress
Deadline



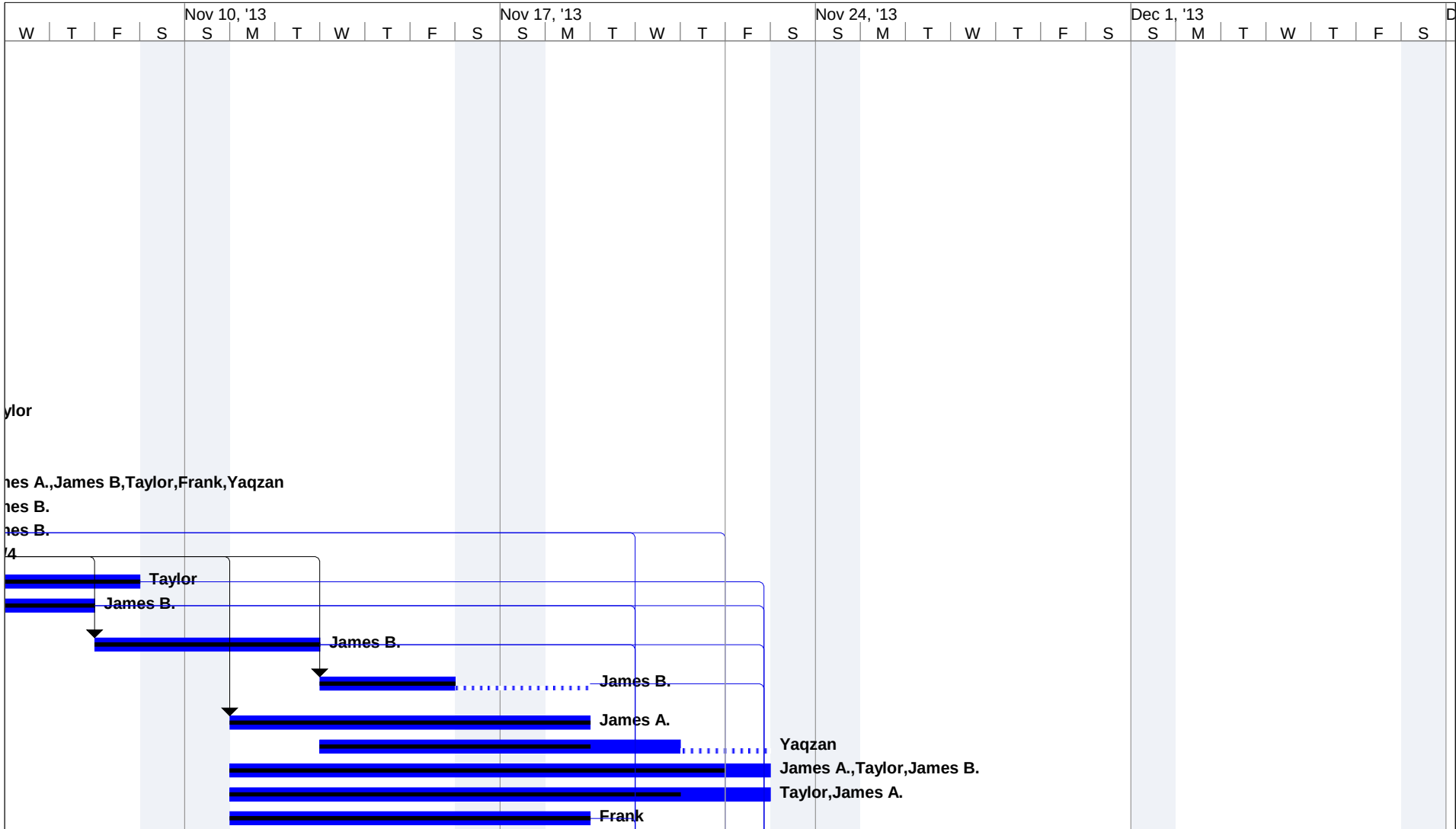
Project: Project1
Date: Fri 11/22/13

Task		Inactive Milestone		Finish-only	
Split		Inactive Summary		External Tasks	
Milestone		Manual Task		External Milestone	
Summary		Duration-only		Progress	
Project Summary		Manual Summary Rollup		Deadline	
External Tasks		Manual Summary			
External Milestone		Start-only			



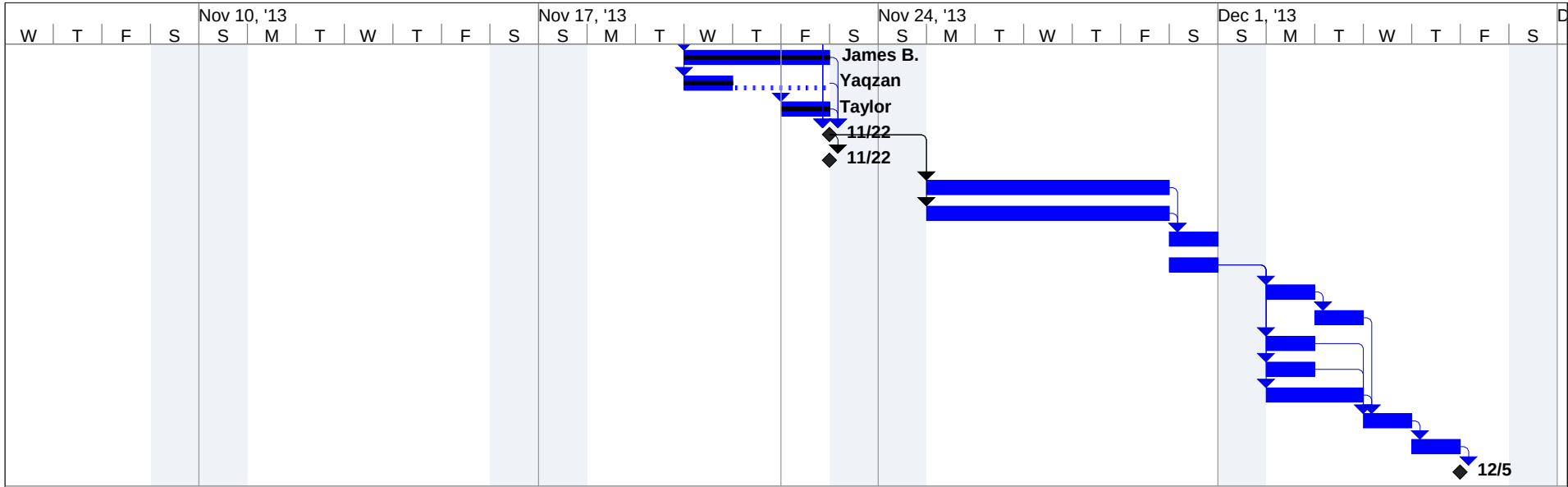
Project: Project1
Date: Fri 11/22/13

Task		Inactive Milestone		Finish-only	
Split		Inactive Summary		External Tasks	
Milestone		Manual Task		External Milestone	
Summary		Duration-only		Progress	
Project Summary		Manual Summary Rollup		Deadline	
External Tasks		Manual Summary			
External Milestone		Start-only			



Project: Project1
Date: Fri 11/22/13

Task		Inactive Milestone		Finish-only	
Split		Inactive Summary		External Tasks	
Milestone		Manual Task		External Milestone	
Summary		Duration-only		Progress	
Project Summary		Manual Summary Rollup		Deadline	
External Tasks		Manual Summary			
External Milestone		Start-only			



Project: Project1
Date: Fri 11/22/13

Task		Inactive Milestone		Finish-only	
Split		Inactive Summary		External Tasks	
Milestone		Manual Task		External Milestone	
Summary		Duration-only		Progress	
Project Summary		Manual Summary Rollup		Deadline	
External Tasks		Manual Summary			
External Milestone		Start-only			