Choice of method

We chose to go with mixture of XP and SCRUM. Scrum is an excellent way to design working days and XP gives great guidlines on how to make code as high quality as possible. Also every one of our team members said that they are at most level 2 developer, so we could not go with even more agile development method like kanban. We chose scrum because of many reasons. First, scrum is implemented in sprints, meaning that at the beginning of every iteration we take items from product backlog and put them into sprint backlog, then split tasks and make sure that everyone works in the iteration. Our iterations are 1 week each (sprint 0 and 1 are two weeks). Second, daily stand up meeting are part of Scrum. Every day we meet, we start with a standard daily meeting where we discuss what has been done, what will be done (on that day) and what problems may occur. After solving all issues and making sure that every person has a taks to do on that day, we start working.

As far as coding goes, we chose to go with Extreme programming development model. XP consists of many useful practises the we can use. At first we start with planning game. We create user stories in close cooperation with the product owner. Then we will make the most simple program that is only capable of performing most simple tasks. For example, no error handling, no exceptions and no input checks will be put into first design. Incremental delivery is another thing we take from XP. We wil make small and frequent releases, so progress could be seen easier. Third practise we use, is pair programming. We will not do pair programming from beginning to end, but only for more complext parts of code. It is useful, because two people can come up with more ideas for solving a specific problem than one person and after creating required part of code it is easier to explain created code to other teammates as well. We will also try out test driven development. As with pair programming, we will not use TDD for very simple parts, but for more complex methods. TDD helps prevent issues when merging parts of code, it also helps save time. If all tests pass before merging, but do not pass after merging, then the part that gives errors is quite clearly distinguishable.

Plan driven vs Agile

There are two main ways to develop code- plan driven and agile method. They differ from each other in many ways. Plan driven approach is good for beginners and small programs. Some Plan driven methods are UP and waterfall. In plan driven approach all tasks are supposed to be fixed from the beginning. It gives clear overview of what needs to be done to reach required result. It is good for beginners, because programmer does not have to come up with a way to solve a problem, programmer has to just follow instructions. Plan driven approach is not good for big projects where changes in requirements might change. It is difficult to implement changes when everything is already written down and the whole workflow is in place. Tests are usually done at the end, when code has been fully developed. This approach does not really give room for errors, especially if estimations about project are not extremely accurate. That is why people came up with agile development methods- to fix issues with implementing changes to program, eliminating need to know all requirements from the beginning and to give programmers more freedom on choosing the way they make part of code work. Some Agilve develompent methods include SCRUM, Kanban and XP. To fix issues regarding changes in program, people came up with incremental delivery system. For example, in Scrum iterations have fixed tasks, but if changes in requirements change, then tasks for next iteration can be changed. In Kanban changes are dealt with even faster- instead of having to wait the end of iteration, new task can be put onto kanban board whenever. Agile development method is great for more professional developers. Since all requirements might not be know from the beginning and tasks may change at any point, it is necessary to be able to adapt to changes and come up with new ways to develop code. There are some problems with agilve approach as well. Since changes may occur, it is difficult to make estimations about the project- fixed price projects are a challenge. Agile development methods also tend towards close communication and cooperation between team and customer. If there are many customers, requirements might not be as clear as they should be. This also might result in having to implement even more changes to existing project as time goes by.