Project Name	...
Online team meeting	https://fau.zoom.us/j/68301268040?pwd=SjUzN3JJRnk3ME0zQ1pjODQ5cFhGQT09
Production system (if any)	...
Test system (if any)	...
GitHub repository	https://github.com/amosproj/amos2022ss08-openid-connect-doctor
GitHub kanban board (project)	- https://github.com/amosproj/amos2022ss08-openid-connect-doctor/projects/1
Team T-shirt (white)	https://www.shirtinator.de/loadBasket/7LOMR46PapD
Team T-shirt (black)	-
Additional materials	...

Last Name	First Name	GitHub User Name	Email Address
Zuber	Yannick	zuberman35	yannick.zuber@fau.de
Arava	Raghunandan	raghunandanarava	raghunandan.arava@fau.de
Kupfer	Michael	FlinkbaumFAU	michael.kupfer@fau.de
Kriesch	Sarah Julia	skriesch	sarah.j.kriesch@fau.de
Rebbe	Philip	prebbe	philip.rebbe@fau.de
Kielburger	Alexander	mindtheme	alexander.kielburger@fau.de
Bilohan	Anna	AnnaBilo	anna.bilohan@fau.de
~~Tavakol~~	~~Mohammad Reza~~	~~moreta-tvkl~~	~~reza.tavakol@fau.de~~
Muktadir	Md Golam	RumiAust	golam.muktadir@fau.de

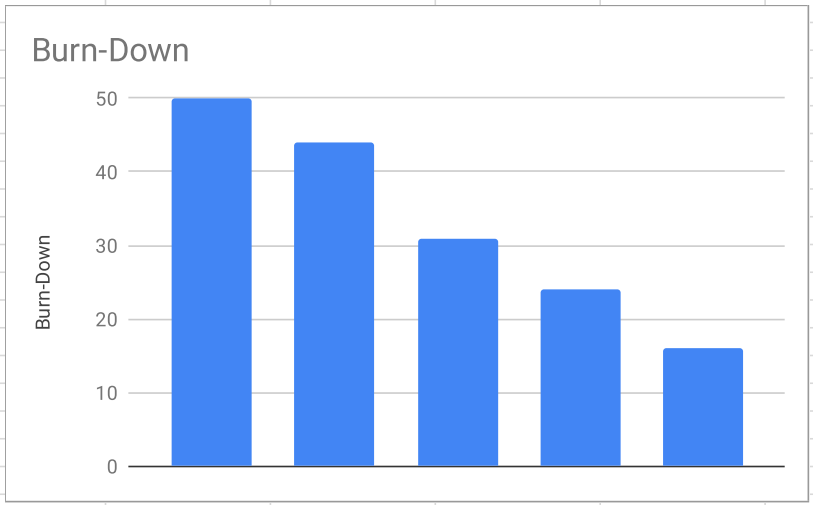
Goals	Keep your meetings short, constructive and simple
Meeting norms	Be on time and notify about not attending at least 4 hours before. Do not interrupt each other. Plan meetings in advance
Working norms	Respect other contributions. Follow defined coding standards, especially to maintain no conflicts between branches
Coordination norms	Be on time. Provide feedback related to code review within the stipulated time.
Communication norms	Being friendly. Do not interrupt each other. Give and accept constructive feedback. Communication via Teams (a backup chat in Signal). Messages in Teams to be checked at least once a day (Mo - Fri). For every voting in MS Teams there should be a resonable deadline (if someone fails to vote before the deadline their votes are not taken into consideration)
Consideration norms	Do not judge others. Everybody voice matters
Cont. improvement norms	Learn from each other. Accept feedback. Test and review your code before submitting it. Track well-being and track feedback
Rewards	Monthly Release Party. Recognition for new features with icons/likes on Github
Sanctions	Whoever is joining the meeting 3 times too late has to buy a beer for everyone
Yannick, Raghunandan, Anna, Golam Muktadir, Philip, Alexander, Sarah,Reza,Michael	

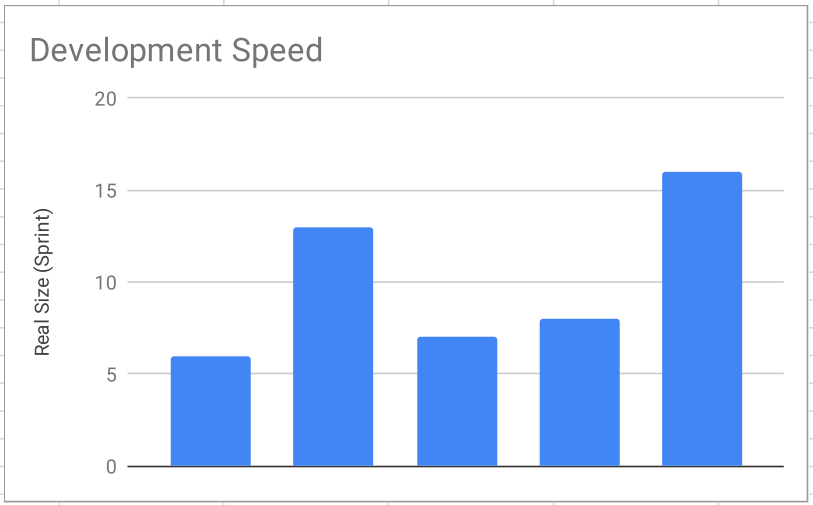
#	Meeting Day	Uni	Comment	Product Owner	Software Developer	Release Manager	Scrum Master
1	2022-04-27			Anna Bilohan / Alexander Kielburger	Everyone else	N/A	Yannick Zuber
2	2022-05-04			Alexander Kielburger	Everyone else	N/A	Yannick Zuber
3	2022-05-11	Yes		Anna Bilohan	Everyone else	N/A	Yannick Zuber
4	2022-05-18			Alexander Kielburger	Everyone else	Sarah Julia Kriesch	Yannick Zuber
5	2022-05-25	Yes		Anna Bilohan	Everyone else	Philip Rebbe	Yannick Zuber
6	2022-06-01			Alexander Kielburger	Everyone else	Raghunandan Arava	Yannick Zuber
7	2022-06-08	Yes	Mid-term due	Anna Bilohan	Everyone else	Michael Kupfer	Yannick Zuber
8	2022-06-15			Alexander Kielburger	Everyone else	Raghunandan Arava	Yannick Zuber
9	2022-06-22			Anna Bilohan	Everyone else	Philip Rebbe	Yannick Zuber
10	2022-01-13	Yes		Alexander Kielburger	Everyone else	Raghunandan Arava	Yannick Zuber
11	2022-01-20			Anna Bilohan	Everyone else	Md Golam Muktadir	Yannick Zuber
12	2022-01-27			Alexander Kielburger	Everyone else	Sarah Julia Kriesch	Yannick Zuber
13	2022-02-03	Yes		Anna Bilohan	Everyone else	Md Golam Muktadir	Yannick Zuber
14	2022-02-10		Demo day!	Alexander Kielburger	Everyone else	Michael Kupfer	Yannick Zuber
15	2022-02-17		Retrospective	Anna Bilohan	Everyone else	N/A	Yannick Zuber

Product Vision	Project Mission
The vision of this project is a tool, that allows a user to automatically analyze and document the tokens an OpenID Connect (OIDC) endpoint provides. By providing this analysis we enable our users (Software Developers, Engineers, etc.) to configure other applications to use the OIDC endpoint. Since we want to ensure the token's usability, the analysis of the tokens will also include a mechanism capable of validating the returned token. Our tool will be independent of any OIDC endpoint providers so that the users have a lot of freedom and are able to provide any OpenID Connect endpoint of their choice.	Our ambitious goal is to develop a fast and easy-to-use tool that is able to provide the users with the analysis of the OpenID Connect tokens and thus allows them to document the endpoint and token structure. Core functionality will be endpoint analysis, requesting tokens, token analysis and providing a short summary to the user.

Term	Definition
Access token	A token that contains the security credentials for a login session and identifies the user, the user's groups, the user's privileges, and, in some cases, a particular application.
Identity provider	A system entity that creates, maintains, and manages identity information for principals and also provides authentication services to relying applications within a federation or distributed network.
JWE	stands for JSON Web Encryption.
JSON Web Encryption	A means of representing encrypted content using JSON data structures.
JWS	JSON Web Signature.
JSON Web Signature	A means of representing signed content using JSON data structures.
JWT	Stands for JSON Web Token,
JSON Web Token	An open standard (RFC 7519) that defines a compact and self-contained way for securely transmitting information between parties as a JSON object.
OIDC	stands for OpenID Connect.
OpenID Connect	An identity layer built on top of the OAuth 2.0 framework. It allows third-party applications to verify the identity of the end-user and to obtain basic user profile information.
Public key	A cryptographic key that can be obtained and used by anyone to encrypt messages intended for a particular recipient, such that the encrypted messages can be deciphered only by using a second key that is known only to the recipient (the private key).
Token	An object (in software or in hardware) which depending on its type represents the right to perform some operation.

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down
1	**Initial Setup**	**Project Setup**						
			Agree on team contract					
			Design team logo					
			Design team T-shirt					
			First meeting with industry partner					
2	**Research**	**Aquire knowledge required for this project**			**8**		**6**	**50**
			# 1 Research the required information to get a token	3		3		
			# 3 Research how to decode a token	5		3		
3	**Planning & First Implementation**	**Finalize the project architecture and start with the implementation**			**13**		**13**	**44**
			# 2 Research how to request a token from an access identity provider	3		3		
			# 4 Create Plan for Implementation	3		3		
			# 5 Enter information	2		2		
			# 7 Agree on language and framework	2		2		
			# 8 Check that an access identity provider exists	3		3		
4	**Testing & Requesting Tokens**	**Initialize a working test-environment and implement token request and endpoint check**			**8**		**7**	**31**
			# 17 Setup a local identity provider for tests	5		5		
			# 21 Implement logic to check that an access identity provider exisits	3		2		
5	**Endpoint analysis**	**Request & Print Endpoint**			**10**		**8**	**24**
			# 6 Get an access token from an access identity provider	5		3		
			# 22 Display OpenId Connect Endpoint	2		2		
			# 23 Filter OpenId Connect Endpoint Display	3		3		
6	**Protocol & Validation**	**Add Protocol, check endpoint structure and validate signature**			**18**		**16**	**16**
			# 9 Decode a JWT token	5		3		
			# 24 Display a token	2		2		
			# 40 Use JSON schema to validate endpoint structure	8		8		
			# 44 Create modules and organize code	3		3		

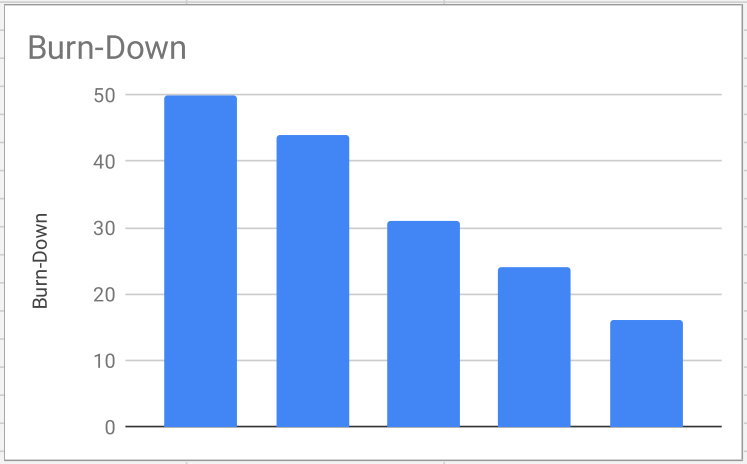

Burn-Down
Burn-Down
50
40
30
20
10
0

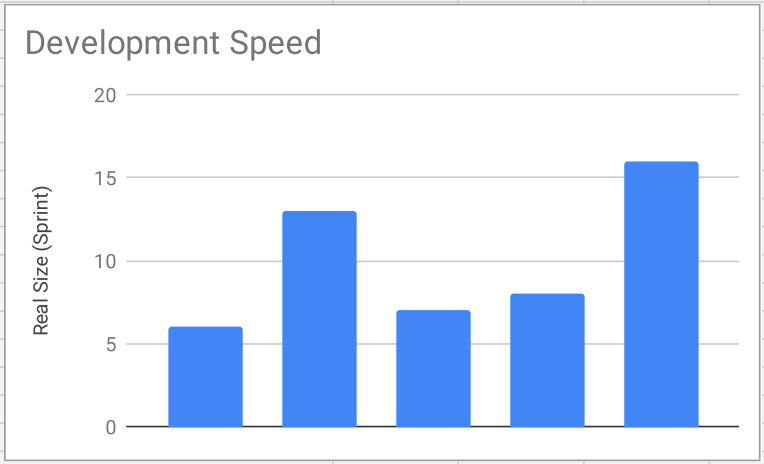

Development Speed
Real Size (Sprint)
20
15
10
5
0

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down
1	Initial Setup	Project Setup						
			Agree on team contract					
			Design team logo					
			Design team T-shirt					
			First meeting with industry partner					
2	Research	Aquire knowledge required for this project			8		6	50
			# 1 Research the required information to get a token	3		3		
			# 3 Research how to decode a token	5		3		
3	Planning & First Implementation	Finalize the project architecture and start wiht the implementation			13		13	44
			# 2 Research how to request a token from an access identity provider	3		3		
			# 4 Create Plan for Implementation	3		3		
			# 5 Enter information	2		2		
			# 7 Agree on language and framework	2		2		
			# 8 Check that an access identity provider exists	3		3		
4	Testing & Requesting Tokens	Initialize a working test-environment and implement token request and endpoint check			8		7	31
			# 17 Setup a local identity provider for tests	5		5		
			# 21 Implement logic to check that an access identity provider exisits	3		2		
5	Endpoint analysis	Request & Print Endpoint			10		8	24
			# 6 Get an access token from an access identity provider	5		3		
			# 22 Display OpenId Connect Endpoint	2		2		
			# 23 Filter OpenId Connect Endpoint Display	3		3		
6	Protocol & Validation	Add Protocol, check endpoint structure and validate signature			18		16	16
			# 9 Decode a JWT token	5		3		
			# 24 Display a token	2		2		
			# 40 Use JSON schema to validate endpoint structure	8		8		
			# 44 Create modules and organize code	3		3		
7	Token Validation	Validate a tokens signature and structure			18		0	
			# 11 Validate Signature	3				
			# 26 Use JSON schema to validate the token structure	5				
			# 34 Get the public key from a file or the identity provider	5				
			# 35 Include binary in the release	5				
			# 39 Add a protocol to the application	5				

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down
8	**Combine Functionality**	**Extend and combine the applications functionality**						
			# 27 Write the Output to File					
			# 28 Implement Password Grant					
			# 55 Provide default setting					
			# 56 Decode token, even it is expired					
			# 59 Combine existing Components					
9	**Core Functionality**	**Finish Core functionality**						
			# 25 Strip a token of non required output					
			# 29 Implement Authorisation Code Flow					
			# 54 Autostart browser on application start					
10	**Frontend Improvements**	**Enhance the usability of the frontend**						
			Replace Parameter List with Drop Down Menu					
			# 50 Move the "Please input provider url" text to the right place					
			# 51 Clicking on the label of a Parameter List entry should also switch the checkbox					
11	**Refinement of the Output**	**Refine the output to improve functionality**						
12	**Refactoring & Demo Day Prep**	**Prepare for the Demo Day and refactor the code (if need be)**						
13	**Finishing the Project**	**Finalize documentation & project**						

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down





#	Feature Definition of Done	Sprint Release Definition of Done	Project Release Definition of Done
	Code has been tested	All issues have been done or went back into the backlog	Core functionality is implemented (token is requested, validated and analysed)
	Optional Logging has been added	All issues that are done, have a real size estimate	Potential additional functionality is added
	No errors (syntax, unit tests, etc.)	Backlog is up to date	No critical bugs
	Peer reviewed by at least one additional person	Release candidate is released, if it is accepted by the PO	Build and deployement documentation is written
	Acceptance criteria met	Code documentation is up to date	Code documentation is written
	The code was merged into the main branch via pull request	The current state of the project has been demo-ed at the meeting	User documentation is ready
		All of the licences are listed in the bill of materials	All of the licences are listed in the bill of materials
			Demo approved by the team

Type	Link / reference
Plan for the implementation	https://docs.google.com/document/d/1JeEaUyx6SOu0O704aXksDRgOUanq58UDLOIG2WYlQgg/edit?usp=sharing

\	Context	Name	Version	License	Comment
		NodeJS	16.15.0	MIT	
	Source-Code	nest-js	8.2.5	MIT	Base-Framework of the application
	Source-Code	panva/jose	27.5.0	MIT	NodeJS-Library to decode and validate a JWT
	Source-Code	panva/node-openid-client	5.1.6	MIT	NodeJS-Library to handle OpenId-Connect-flows
	Source-Code	ajv-validator/ajv	8.9.0	MIT	NodeJS-Library to check JSON schema
	Source-Code	node-jsonwebtoken	8.5.1	MIT	Node-JS library for the validation of signatures

Last Name	First Name	Value					
				4.40	NOK		
Arava	**Raghunandan**	5					
Kupfer	**Michael**	5					
Kriesch	**Sarah Julia**	2		0	No size		
Rebbe	**Philip**	5		1	Trivial size		
				2	Small size		
				3	Medium size		
Tavakol	**Mohammad Reza**			5	Large size		
Muktadir	**Md Golam**	5		8	Very large size		
				13	Too large (size)		