This section outlines the proposed system IU-SmartCert, a blockchain-based educational credential management system with a selective disclosure option. Our system has 3 main groups of users: issuers, learners, and relying parties. An issuer can be broadly understood to include individuals or organizations qualified to issue a credential. A learner is a person who received the issued credential. A relying party in this paper is used to refer to a person or a company who uses and wants to check the validity of a credential. Figure 1 shows key functions and typical interactions among users of our system. First, given a set of credentials to issue, the issuer organizes each credential into our proposed format that composes of two parts: a mandatory component and a list of optional components, then inputs the fingerprint of all credentials to IU-SmartCert. From the input fingerprints, the system constructs a Merkle tree and publishes the root node of the tree along with the issuer identity to the public Ethereum blockchain as a smart contract. After publishing, the issuer sends a digital credential and its receipt to each learner. Later when the learner needs to show her/his credential to a relying party, like an employer, s/he can select the most relevant optional components of the credential to present along with the mandatory component to the employer. Finally, a relying party verifies the authenticity and the integrity of a given credential by checking the received data against the public key infrastructure and the Ethereum public blockchain. The verification process can be done independently without contacting the issuer, and even without using the IU-SmartCert system. Formally, our blockchain-based credential management system provides functions to
R1. Define and issue credentials with a mandatory component and optional components for selective sharing,
R2. Verify and validate a credential,
R3. Select optional components of the credential to disclose,
R4. Revoke a credential issued by the system In addition, to increase user’s flexibility and initiative, the system should have the following quality attributes:
R5. Ability to independently check the integrity and validity of a credential
R6. Security for credentials. In the following sections, we describe in detail important procedures in the IU-SmartCert system.

3.1 Defining a Credential Schema

In IU-SmartCert, issuers of credentials like universities and institutions can define their own schema and vocabularies of a credential for each program. A credential in IU-SmartCert composes of two parts: - One mandatory component - and a list of optional components A mandatory component is one that learner must disclose to every replying party. This component stores all information needed to validate and evaluate a credential. For instance, in a credential of an undergraduate student, the mandatory component is the diploma which contains the name of the university, the name of the learner, the issued date, and the enrollment year. The list of optional components is a tool for issuers to define a flexible credential data model. An issuer can issue credentials with different levels of granularity, therefore allow learners to selectively disclose their credential on demand. For instance, a university could issue a transcript of a student by issuing the score of each course separately as a list of optional components. This credential schema allows the student to choose courses to disclose to a relying party. Conversely, if a university does not allow students to cherry-pick courses to disclose, the university issue the entire transcript as a single component of the credential. Moreover, in a stricter rule that requires students to disclose the transcript along with their diploma, the university can combine the diploma and the transcript 3.1 Defining a Credential Schema In IU-SmartCert, issuers of credentials like universities and institutions can define their own schema and vocabularies of a credential for each program. A credential in IU-SmartCert composes of two parts: - One mandatory component - and a list of optional components A mandatory component is one that learner must disclose to every replying party. This component stores all information needed to validate and evaluate a credential. For instance, in a credential of an undergraduate student, the mandatory component is the diploma which contains the name of the university, the name of the learner, the issued date, and the enrollment year. The list of optional components is a tool for issuers to define a flexible credential data model. An issuer can issue credentials with different levels of granularity, therefore allow learners to selectively disclose their credential on demand. For instance, a university could issue a transcript of a student by issuing the score of each course separately as a list of optional components. This credential schema allows the student to choose courses to disclose to a relying party. Conversely, if a university does not allow students to cherry-pick courses to disclose, the university issue the entire transcript as a single component of the credential. Moreover, in a stricter rule that requires students to disclose the transcript along with their diploma, the university can combine the diploma and the transcript into a single mandatory component in the input to the IU-SmartCert. Therefore, the structure helps to fulfill the requirement R1 of the system. The procedure to define a schema and vocabularies in IU-SmartCert is performed via the data input of an issuer. For every credential, the issuer must provide a list of files corresponding to each component in the credential with the following naming rule CredentialID.ComponentName.(R).ext where - CredentialID is the identity of a credential and all components of a credential will share the same value. - ComponentName is the name of a component. - .(R) is the marker for the mandatory component. An optional component's filename does not include this marker. - .ext is the file extension. For instance, consider a bachelor's credential composes of a diploma, a transcript and a scientific profile. The issuer could define that the diploma is a mandatory component, while the transcript and the scientific profile are optional components. In such case, the input to the IU-SmartCert of a credential with identifier ITIU01 is 3 pdf files named as following IUIT01.diploma.(R).pdf IUIT01.transcript.pdf IUIT01.profile.pdf

3.2 Issuing Credentials

When learners complete a program, the institution generates digital credentials of learners and passes them to the IU-SmartCert system. The system then organizes and publishes the digital fingerprint of those credentials to the Ethereum blockchain in such a way that learners can choose components of their credential to disclose and a relying party can validate a given credential (Fig. 2).

Constructing Merkle Tree. To begin this procedure, we use the schema and the digital fingerprint of each credential to construct a Merkle tree [13]. We first combine the identity of the credential with the content of the component and its type, mandatory or optional, defined by the schema, then uses SHA-256 to calculate the hash value of the combination and create a leaf node in the Merkle tree. We apply this procedure to all components of all input credentials and obtain corresponding leaf nodes. From those leaf nodes, we build the rest of the tree by concatenating two leaf nodes, calculating a new hash value resulting in a parent node, and continuing up toward the root of the Merkle tree. We can construct a Merkle tree from any arbitrary number of credentials and always results in a single root node, so an institution could issue credentials in batch to save time and effort.

Publishing Data as a Smart Contract.
Once the Merkle tree is built, the IU-SmartCert system publishes the hash value of the root node and supporting data to the Ethereum blockchain as a smart contract so that a relying party can validate an issued credential independently without contacting the issuer. The smart contract (as shown in Fig. 3) consists of - institute A read-only variable for the hash of the issuer name. The issuer name is the organization field in the issuer's X.509 certificate, and thus binds to the identity of the issuer. - MTRoot A read-only variable for the hash of the root of the Merkle tree. - revocationList A list of revoked credentials along with a reason of the revocation. The function is accessible only for the issuer which is the owner of the contract. See Sect. 3.5 for the revoking procedure. - verify(bytes32[], bytes32) A function to check whether a component belongs to the Merkle tree represented by the root node stored in MTRoot of the contract. - isValid(bytes32) A function to check whether a credential is revoked. If the credential is revoked, the function returns false with a reason, otherwise, returns true. - revoke(bytes32, string) A function to revoke a credential. See Sect. 3.5 for the revoking procedure. It is worth noting that the smart contract stores only the root node of the Merkle tree constructed from the batch of credentials, and no other data about the credentials are published. When the deployment of the smart contract is confirmed, the system keeps the metadata to generate receipts for learners.

3.3 Generating Receipts

After publishing credentials to Ethereum network, the system generates a receipt for each credential and sends it along with the digital credential to the corresponding learner. A relying party will use it to verify and validate the credential. A receipt contains metadata of the smart contract, a proof of existence in the Merkle tree of each component of the credential, and the X.509 certificate of the issuer. The metadata in a receipt is to help a relying party to verify the identity of the issuer and the validity of the smart contract. The metadata consists of the address of the smart contract that manages the credential, the hash value of the transaction that deployed the smart contract and the identity of the issuer. To provide a verifiable identity, we use a hybrid approach [6] where we establish a binding between accounts in Ethereum and an X.509 public key certificate of the issuer. This X.509 certificate is issued by a trusted certificate authority (CA) of the traditional public key infrastructure (PKI), which comprises of hardware and software for creating, managing, distributing, using and revoking public keys. At registration time, CA verifies the identity of the issuer and writes its information into the X.509 certificate; then, CA digitally signs this certificate with its private key [1]. Therefore, originality of digital credentials in this system is guaranteed. Before using our system IU-SmartCert, the issuer needs to endorse the Ethereum account used to publish credentials by signing the respective address with the private key of the X.509 certificate. Then, the issuer input to the system the address of its Ethereum account, the signature, and its X.509 certificate chain. Later, replying parties like employers can retrieve from the receipt the information to verify them versus their trusted certificate authority, thereby authenticating the identity of the issuer and the validity of the smart contract. In a receipt, the proof of existence of a component is extracted from the Merkle tree build in the issuing phase. Since each component corresponds to a leaf node of the tree, its proof of existence is a list of nodes in the Merkle tree required to recalculate the path from the corresponding leaf node to the root node [13]. In an example in Fig. 4, the proof of existence of component 3 is the hash value Hash 4 and the hash value Hash 12. From these proofs, one can calculate the path from component 3 to the root node.

Finally, the system generates one receipt for each credential as a file in JSON data format (as shown in Fig. 5) with the following fields: - issuedOn: the time when the credential was published to the blockchain - transactionHash: the hash value of the transaction that deployed the smart contract that manages the credential - contractAddress: the address of the smart contract that manages the credential - credentialID: the identity of the credential - components: the data to prove the authenticity of the credential. For each component of the credential, the data includes • name: the name of the component • mandatory: a boolean value indicating whether the component is mandatory or not • proof: the proof of existence of the component in the Merkle tree • hash: a hash value of the concatenation of the credential's identity, the type, and the content of the component. - issuer: the verifiable identity of the issuer • ethereumAccount: the Ethereum account of the issuer • ethereumAccountSignature: the signature of the issuer's Ethereum account endorsed by the private key of the X.509 certificate • IssuerCertificateChain: the chain of the X.509 certificates of the issuer in PEM format

3.4 Exchanging Credentials

In our proposed system, learners can exchange their credentials with a selective disclosure option. In other words, learners can choose components to share, and also can choose components not to share while following the schema defined by issuers (Requirement R3). The exchanging credentials begin when an employer requests the learner to present his credential. First, the learner takes his credential and picks the most relevant components to share. This selection is possible because credentials in our system are organized into two parts a mandatory component and a list of optional components. And since the type of each component is defined by the issuer, the learner can freely pick and skip some optional components without invalidating the credential. After the selection, the learner needs to generate a new receipt which is the proof of existence for those selected components. He can upload his original credential’s receipt to the IU-SmartCert system, select the chosen components, and download a new receipt. On the other hand, the learner can make the new receipt on his own without using the system. He can open his receipt in a text editor, remove the sections corresponding to the components that are not chosen, and save the file as a new receipt. For example, in Fig. 5 there are two valid receipts of one credential. The receipt on the left is used when the learner would like to show all of the three components of his credential. The receipt in the right is used when the learner chooses not to share the ScienceProfile component. Finally, the learner sends the new receipt and related files to the employer.

3.5 Revoking Credentials

Like other credential management systems, the IU-SmartCert allows issuers to revoke issued credentials (Requirement R4). While in IU-SmartCert learners can select components to present to relying parties, they always have to present the mandatory component of the credential otherwise the credential is invalid. With that structure, issuers to revoke an issued credential by marking its mandatory component as revoked. In detail, we store in the smart contract that manages the issued credentials a list of revoked components along with the reason for their revocation. And only the owner of the smart contract, i.e. the issuer of the credential, can add a record to that revocation list. Later, any relying party can check the status of a credential by checking its mandatory component against the revocation list. In Fig. 3 the data and the functions for the revocation are the revocationList, the revoke, and the isValid functions.

3.6 Verifying a Credential

An employer or any relying party can verify a credential issued by IU-SmartCert from the files, the receipt and the Ethereum network without contacting the issuing institution. The credential verifying process includes the following steps:
1. Check the receipt information and the issuer’s information - The issuer’s X.509 certificate - The signature of the Ethereum account used to issue the credential. - Validity of the owner of the smart contract on Ethereum. - The name of the issuer in the smart contract and the name on the X.509 certificate.
2. Check the integrity of all components of the credential. This step checks the number, the type and the hash value of the files against the value stored on the receipt.
3. Check the validity of the components in the certificate. This step checks the hash value and proofs to confirm whether the credential belongs to the merkle tree whose root node is published on the Ethereum blockchain.
4. Check whether the certificate is revoked. This step checks if the mandatory component of the credential in the list of revoked credentials stored in the smart contract on the Ethereum blockchain.

The above procedure is implemented in IU-SmartCert with a progress bar for each step as shown in Fig. 6. On the other hand, relying parties can perform the verifying procedure on their own. Once received a credential and its receipt, a relying party can extract all the data to verify a credential without the involvement of the issuer. Then with a connection to the Ethereum blockchain and any publicly available tools to verify a digital signature and a X.509 certificate, the relying party can finish the verification procedure. That helps us fulfill the requirements R2 and R5.